

UNIVERSITY OF SOUTHAMPTON

APPLYING GAMIFICATION TO TEACHING CYBER SECURITY

BY

REECE BUCKLE

PROJECT SUPERVISOR: DR NAWFAL FADHEL

SECOND EXAMINER: DR ANDREW SOGOKON

A PROJECT REPORT SUBMITTED FOR THE AWARD OF BSC
COMPUTER SCIENCE

DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

MAY 2021

Abstract

Due to the ever changing landscape of technology and cyber security, there is a requirement to promote cyber security awareness where possible. Typically this is done via government endorsed training schemes, website campaigns, fliers and posters and gamified strategies. As a strategy towards this problem, this report investigates the use of gamification mechanics in order to teach cyber security concepts in a fun and interactive way. Furthermore, despite many serious cyber security games being developed for Universities and businesses, there are a lack of cyber security games that are also relevant and available to the general public. Therefore, this project will aspire to develop a multiplayer board game in which cyber security vulnerabilities are illustrated through mini-game challenges, cyber attack & defence cards and a contextually appropriate setting. The goal of this project is to produce a game which is relevant for both businesses and the public domain - and ultimately leaves the player more conscientious in regards to how they interact with technology.

Statement of Originality

I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.

I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.

I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

I have acknowledged all sources, and identified any content taken from elsewhere.

I have not used any resources produced by anyone else.

I have used some royalty-free assets for my game (few sound affects, the tile pattern, 3D Unit Models and the UI icons. These are acknowledged during the report of mention, as well as in the acknowledgements of my game - should I make it doubly clear here as well?

I did all the work myself, or with my allocated group, and have not helped anyone else.

The material in the report is genuine, and I have included all my data/-code/designs.

I have not submitted any part of this work for another assessment.

Contents

1	Introduction	5
1.1	Goals	5
1.2	Scope	6
2	Literature Review	7
2.1	The Problem with Cyber Security Training Programs	7
2.2	Why Use a Game-Based Learning Approach?	8
2.3	Appropriate Gamification Mechanics	8
2.4	A Review of Pre-Existing Cyber Security Games	9
2.5	An Analysis of Games Reviewed	10
2.6	Related Work	10
2.6.1	OWASP Cornucopia Game	10
2.6.2	Control Alt Hack	11
2.6.3	Program Wars	11
2.6.4	Riskio	12
2.7	A Solution to the Problem Statement	13
2.8	Conclusion	14
3	Requirements	15
3.1	Stakeholder - Personas	15
3.2	Functional Requirements	17
3.3	Non-Functional Requirements	17
3.4	User Stories	18
3.5	Constraints and Limitations	19
4	Design	20
4.1	Prototyping the Application	20
4.2	Applying the MDA Framework	20
4.2.1	Mechanics	21
4.2.2	Dynamics	22
4.2.3	Aesthetics	22
4.3	Mapping the OWASP Top 10 to Game Mechanics	23
4.4	UML Modelling - A Class Diagram Representation	24
4.5	Mapping the MDA Framework to Class Functions	25
4.6	UML Modelling - Activity Diagram	27
4.7	Visual Design, Contrast and Accessibility	28
4.8	Colour Blindness	29
4.9	Designing for the Web and Mobile	30
4.10	Design Patterns for Unity & C#	31
4.10.1	Component Design Pattern	31
4.10.2	Singleton Design Pattern	32
4.10.3	Flyweight Design Pattern	32
5	Implementation	33

5.1	Multiplayer with Photon Cloud	33
5.2	Photon Unity Networking 2 Architecture	34
5.3	Remote Procedure Calls - RPCs	34
5.4	Opening the Application	35
5.5	Implementation of Unit Movement	36
5.6	Movement Mechanics - Breadth First Search	37
5.7	Ability Range	41
5.8	Implementation of Units and Unit Abilities	42
5.8.1	Implementation of Scout Unit	42
5.8.2	Implementation of Hacker Unit	43
5.8.3	Implementation of Heavy Unit	44
5.8.4	Implementation of Analyst Unit	45
5.8.5	Implementation of Server Units	46
5.9	Information Menu	47
5.10	Status, Hints & Tips Bar	47
5.11	History Log	48
5.12	Quit Menu	48
6	Testing and Evaluation	49
6.1	Unit Testing	49
6.2	Identifying Edge Cases	49
6.3	ParallelSync for Live Builds	50
6.4	Performance Testing	50
6.5	Achieving a Target Frame Rate	51
6.6	Evaluation and Optimisation	51
6.7	Evaluation of Networking Strategy	52
6.8	Evaluation of Application Frame Rate	53
6.9	Evaluation of Project Completion	53
7	Project Management	55
7.1	Reflection of Project Management	55
7.2	Gantt Chart - Phase One	56
7.3	Gantt Chart - Phase Two	56
7.4	Risk Assessment	58
8	Future Work	59
9	Conclusion	60
10	Bibliography	62
11	Appendices	67
11.1	Unit Test Cases	67
11.2	Performance Profiling in Unity3D	69

1 Introduction

As modern-day technology is ever evolving, the number of users who interact with technology increases consequently. As a result, the risk of an individual, or business, becoming a victim to cyber-crime increases proportionately. Small and medium-sized businesses (SMBs) are the biggest sectors targeted by cyber-criminals [1], which stem from issues such as budget restraints and expressing a lack of understanding towards cyber security concepts.

In fact, because of COVID-19 changing the dynamic of industry standards this year, a statistical analysis from May 2020 (UK) showed that individuals experiencing targeted hacking increased by 77.41% - in comparison with the previous year [2]. This is most likely since employees are encouraged to work from home via their personal computers. Consequently, this has fed into a new strategy whereby cyber-criminals are moving laterally into organisational infrastructure by targeting and infecting employees at their less secure personal computers [3].

Regarding this problem, this paper will explore the effectiveness of educational games - which has been shown to have an advantage on the learning outcome in comparison with traditional training material [4].

Problem Statement

Despite the existence of many cyber security awareness programs, there is a lack of effective, widespread cyber security awareness

Research Question

Can teaching cyber security through a gamified medium improve cyber security awareness?

Hypothesis

Using the MDA framework will satisfy the educational requirements for creating an serious game

1.1 Goals

The primary goal is to apply gamification mechanics, to a multi-player game, in order to illustrate the OWASP Top 10 security weaknesses. The expected result of this project is to create an accessible, online, multiplayer board game which could be used for both recreational and training purposes.

Application specific goals:

- The application will support 2-person multiplayer
- The application will be accessible on PC, tablets and the web
- The application will be playable cross-platform
- The application will illustrate a range of cyber security attacks and weaknesses

- The application will be extensible to support new scenarios and mechanics

1.2 Scope

As cyber-security is an incredibly broad field, only the OWASP Top 10 will be used to create an educational scenario of defending and attacking web applications. During the literature review, a range of pre-existing online game types are evaluated – with appropriate mechanics extracted and identified from Forde’s research [5]. Using the findings from this evaluation, and the MDA framework [6], an appropriate serious game style is identified – with multiplayer being the core focus. Finally, the application is completed using Unity3D, Photon Unity Networking 2 and Photon Cloud.

2 Literature Review

This section presents the current problem with cyber security training methodologies. From this, gamification mechanics, cyber security games and related work are reviewed to identify a potential solution.

2.1 The Problem with Cyber Security Training Programs

Regarding the delivery of training programs, generalised security advice has little effect on changing the behaviour of employees within SMBs as they fail to relate the information to their workplace-environment [1, 7]. Furthermore, non-scenario-specific training fails to demonstrate how security risks link together in a multifaceted social engineering attack [7]. Lastly, training programs are generally undertaken in a formal setting which fosters a situation of recipients not absorbing the information effectively because they aim to finish the material as fast as possible [7].

To summarise, the 4 core problems for SMBs [1, 8] are:

1. SMBs can be heavily constrained by a limited budget
2. SMBs are difficult to reach because they do not understand the severity of data breaches
3. SMBs are often distracted by the operational requirements for setting up and running a small business
4. SMBs struggle to identify their assets in terms of the risks associated with them

From this, it is evident that a light-weight training application that can be tailored for multiple scenarios would be an appropriate solution for training these employees. Furthermore, delivering the training material through a gamified medium should lead to an increased rate of information absorption - even in a workplace environment.

2.2 Why Use a Game-Based Learning Approach?

Many of the challenges from traditional training programs can be overcome with game-based learning strategies [4]. A summary of motivations for these strategies [4, 9, 10, 11, 12] include:

- Rapid progress paired with instant feedback
- Strong user engagement
- Allows the user to learn from mistakes in a safe, non-punishing environment which would otherwise discourage exploration (due to the fear of failure)
- Encourages self-learning at the user's desired pace
- Deeply immerses the player into the game world whereby learning feels like a secondary objective
- Low cost to produce in comparison to larger training schemes
- Requires little to no supervision
- Easy to distribute across multiple platforms and incorporate into the workplace
- Easy to integrate within events such as hackathons and other cyber security awareness gatherings
- An integrated rewards system such as badges, hidden achievements lead to a desire to win
- Contextually appropriate by incorporating a story that is similar to the real world

2.3 Appropriate Gamification Mechanics

To evaluate pre-existing cyber security games in the next section, Forde's report [5] identifies the following gamification mechanics which increase the adoption rate of positive cyber security trends within the workplace:

- | | | |
|---|---|--|
| <ul style="list-style-type: none">• Avatar & User Profile• Badges & Privileges• Challenge• Competition• Collaboration | <ul style="list-style-type: none">• Feedback & Guidance• Goals & Objectives• Incentives & Rewards• Leader boards• Points System | <ul style="list-style-type: none">• Progress & Levels• Role Playing• Story• Tips & Hints System |
|---|---|--|

2.4 A Review of Pre-Existing Cyber Security Games

Table 1 presents an analysis of educational cyber security games.

Game	Game Type	Description & Key Findings	Target Audience	Mechanics
Game of Threats [13]	Multiplatform (Mobile, PC)	Employees are split into teams of attackers and defenders to simulate scenarios of cyber-attacks and appropriate responses. Demonstrates the mindset of both attackers and defenders. Prompts discussion with colleagues in teams to popularise cyber security readiness	Business Employees	Feedback, Incentives, Competition
Cybersecurity Lab [14]	Web-based Point-and-Click	Player runs a fake business and must complete mini games for the business to succeed. These mini games teach children how to spot phishing emails, construct strong passwords and simple programming principles	Children (aged 7-12)	Tips & Hints, Points System, Achievements, Avatar, Progress, Feedback
Webonauts Internet Academy [15]	Web-based Point-and-Click Side-Scroller	Player is an astronaut in which they rank up their status by demonstrating smart and good behaviour. Teaches children how to be respectful and protective themselves online via website certificates, establishing privacy settings and not giving out passwords.	Children (aged 7-12)	Tips & Hints, Feedback, Avatar, Badges
Targeted Attack [16]	Web-based Point-and-Click	Player is a CEO in a choice-based simulation which teaches employees the correct responses and mitigations to cyber-attacks through trial and error.	Business Employees	Feedback, Challenge, Story
Keep Tradition Secure [17]	Web-based Point-and-Click	Player is a student trying to take down a fictional cybercriminal by making smart cyber security decisions. Teaches the player how to use campus VPN via quizzes.	University Students	Tips & Hints, Feedback, Rewards
Cyber Awareness Challenge [18]	Training Simulation	Player is placed in a simulation of a workplace environment to analyse how they behave, and provide guidance to behave safely and responsibly	Business Employees	Tips & Hints, Points System, Feedback, Story
Cyberland [19]	Web-based Point-and-Click	Cyber Security Challenge UK hosts a variety of mini games (Cyberland), and competitions between schools, universities, businesses and government institutes.	Students (All Levels)	Tips & Hints, Story, Goals, Competition, Feedback
Hacknet [20]	Video Game Point-and-Click	Hacknet is a terminal-based hacking simulator. It teaches the player how to navigate networks, search for hidden files/folders, bypass authorisation and general Unix commands	Gamers	Story, Progress, Achievements, Feedback
Classcraft [21]	Web-based Point-and-Click	Classcraft incorporates gamification principles using project management software. Teachers set goals and challenges within a classroom which encourages teamwork between students. Students can be rewarded and punished for good & bad behaviour.	Students (Lower Education)	Points System, Leader board, Goals, Avatar, Competition, Feedback, Incentives, Badges

Table 1: Analysis of Pre-Existing Cyber Security Games

2.5 An Analysis of Games Reviewed

Many of the educational games reviewed in Table 1 relied heavily on presenting facts and then subsequently quizzing the user. However, users can utilise common sense to rule out incorrect answers. This fails to invoke critical thinking and keeping the user engaged[22]. As a solution, gamified strategies should incorporate a variety of factual, conceptual, and procedural learning methodologies [22]. Conceptual understanding should be prioritised due to security-risks often being linked together in multi-faceted attacks [7].

Of all games reviewed and found, only Game of Threats [13] supported multiplayer which suggests there are a lack of educational, multiplayer games accessible online. According to [23], online, educational, multiplayer games are rare as it is difficult for scenarios to be built and managed as the player number scales. This paper recommends multiple scenarios are kept simple and designed for just 2-4 players as open world games tend to lose focus.

Lastly, every game that was accessible online was targeting students and employees; this suggests a lack in resources available to the public.

2.6 Related Work

The following section reviews literature related to this project's proposed solution of creating a multiplayer tabletop game.

2.6.1 OWASP Cornucopia Game

OWASP Cornucopia [24] is a table top card game which is designed at teaching a secure agile methodology for software development. A deck of cards is mapped from the OWASP Top Vulnerabilities and CAPEC software attack patterns. A core requirement of this game is that it requires an application, module or software development project in order to assess, and the game itself facilitates and sparks conversation about securely developing the game. In a post review study [25], it was found that teams who discussed OWASP principles whilst partaking in this activity expressed more enthusiasm and enjoyment than teams who merely discussed security without the Cornucopia, therefore exemplifying the effectivity of gamifying a scenario.

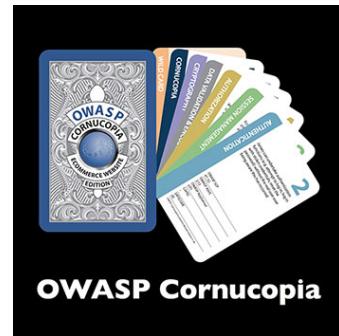


Figure 1: OWASP Cornucopia Deck [24]

2.6.2 Control Alt Hack

Control-Alt-Hack [12] is a physical tabletop card game. The core mechanic instructs players to control ‘white hat’ hacker archetypes to complete missions in which cyber security scenarios are exemplified.

From the feedback gathered in [12], (of 450 students and 150 educators), it was shown that the social engagement from multiplayer scenarios was the biggest contributor that positively impacted security awareness among the participants. However, the main criticisms identified was that there ‘there was not enough educational value which specifically tested their knowledge of vulnerabilities and penetrations techniques’. This is likely due to the lack of strategy in the game, in which a conceptual strategic environment could provide a means to testing knowledge.

2.6.3 Program Wars

Program Wars [11] is a web-based 2-player game in which both players have access to cards related to programming - ‘instruction’, ‘repetition’ and ‘variable’ cards. Overall this game was positively received for enabling people to positive connect concepts explored within the game to an actual programming language. Another note is that because nearly all players were familiar with card games (and games of this nature), and that the mechanics were relatively simple, nearly all of the 46 participants had no difficulty playing the game with a fully implemented tutorial.



Figure 2: Control Alt Hack Game Board [12]

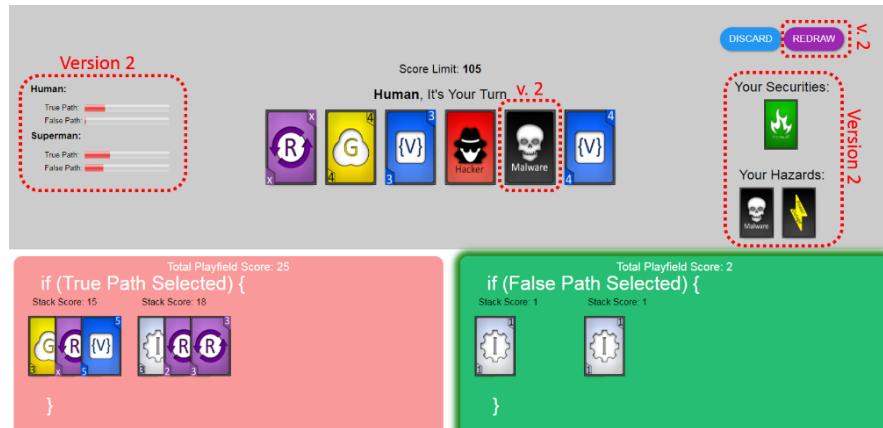


Figure 3: Program Wars Gameplay [11]

2.6.4 Riskio

Riskio [9] is a 3-5 player tabletop card game. The core mechanic is to enable players to identify threats within an office scenario by utilising decks of ‘attack’, ‘information’ and ‘defence’ cards. By utilising a combination of these 3 attack cards, players are encouraged to identify with both sides of a cyber attack scenario, looking into how they can socially engineer a scenario as well as prevent it. Riskio also requires the role of a ‘games master’ with skilled cyber security knowledge in order to facilitate the learning environment. A post review analysis identified this strategy as successful in eliciting a player’s curiosity to read into each card in order to understand the core gameplay. However, it was also identified the importance of a cost function between action mechanics to create meaningful play. This lead to the development of a unit cost ability moves, and a risk to this action point gain if certain units are defeated.

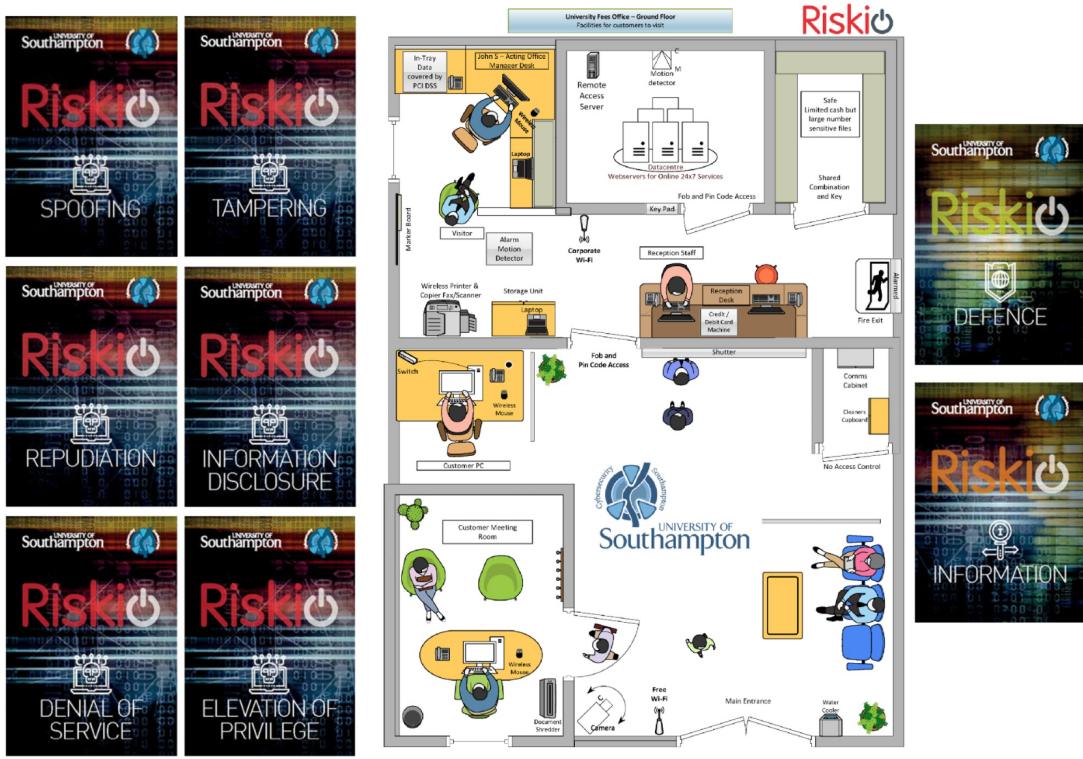


Figure 4: Riskio Cards and Board [9]

2.7 A Solution to the Problem Statement

From reviewing pre-existing games, the following 8 features and gamified mechanics were identified as valuable components when designing an educational game. This project aims to combine these features to provide a solution to the initial problem statement outlined in this report.

Table 2 illustrates the comparison of these features to this project.

- | Feature / Mechanic | | | | | | | | |
|---|--|--|--|--|--|--|--|--|
| 1. Multi-platform - Available on PC, Web and Mobile | | | | | | | | |
| 2. Multiplayer - Supports online matchmaking | | | | | | | | |
| 3. Competition - Hackathons, global challenges or direct multiplayer | | | | | | | | |
| 4. Educational - Incorporates an educational aspect within the game | | | | | | | | |
| 5. Hints & Tips or Feedback - Provided by the application or from a real life 'game master' | | | | | | | | |
| 6. Accessible - Free and readily available online | | | | | | | | |
| 7. Multiple Audiences - Relevant for all age categories and occupations | | | | | | | | |
| 8. Extensible - Flexible to add new scenarios and 'gamify' a subject | | | | | | | | |

Application vs Feature ID	1	2	3	4	5	6	7	8
Game of Threats	✓	✓	✓	✓				✓
Cyber Security Labs	✓			✓	✓	✓		
Webonauts Internet Academy				✓	✓	✓		
Targeted Attack	✓			✓		✓		
Keep Tradition Secure	✓			✓	✓	✓		
Cyber Awareness Challenge				✓	✓	✓	✓	
Cyberland	✓		✓	✓	✓	✓	✓	✓
Hacknet				✓			✓	
Classcraft	✓		✓		✓	✓		✓
This Project		✓	✓	✓	✓	✓	✓	✓

Table 2: Proposed Features for this Project

I could summarise this section directly after the related work and save space, and also remove conclusion if not necessary

Regarding the related work, the following conclusions have been made to ensure we are building a playable project that builds on from previous work as well as being functionally usable

Cost Functions

To prevent the application's mechanics simply representing attacks and defences to security vulnerabilities, the usage of a 'cost function' to the underlying mechanics should add an extra depth of strategy leading to meaningful play.

Familiarity

Regarding Program Wars [11], by developing an application archetypal that users are already familiar with, the application should be easy to understand and adopt. As such the core game-type will be based off a typical turn based board game similar to Chess - which has ancient derivations in all cultures around the world.

Implementing Strategy

Since a major criticism in Control-Alt-Hack [12] was the lack of strategy, designing an application off a game-type such as Chess which is a highly acclaimed strategy game, a ruleset of mechanics similar to this should invoke critical thinking and hopefully test the user's acquired knowledge within the application.

2.8 Conclusion

To summarise, a cyber security training application needs to be cost effective and target the specific needs and requirements of the workplace in order to be viable for SMBs. Where traditional training methodologies fall short, gamified strategies can offer a more cost-effective solution for teaching relevant cyber security concepts. However, many of the educational games reviewed included a number of flaws which stem from lack of gamified mechanics that encourage meaningful play. From Forde's mechanics analysis [5], multiplayer, challenge and competition are key mechanics which are missing in widely available cyber security games. Therefore, the goal is to incorporate these mechanics into an application in which scenarios can be flexibly created for a variety of specific scenarios.

3 Requirements

Based on the findings in the literature review, appropriate personas are constructed from the stakeholders who could benefit from the proposed application. From this, requirements are identified and refined into user stories, with a MoSCoW priority. These user stories are then scheduled into a Gantt Chart for project management.

3.1 Stakeholder - Personas

From analysis of pre-existing cyber security games, a list of stakeholders in which educational cyber security games were developed and targeting includes:

- Students (All Levels)
 - Researchers and Professors
 - Employees (SMBs)
 - Security Advocates
-



Persona One - Olivia - Student

Olivia is an undergraduate Physics student with a strong interest in programming and web development. Although she does not major in Computer Science, she would still like to learn more about the most common web application security risks.

Olivia would also love to educate her parents in basic cyber-security as her own father is not well acquainted with internet safety – and has succumbed to phishing emails himself.

Figure 5: Royalty - Free Image [26]

Because her Dad suffers from colour blindness, her requirements is a colour-blind friendly game in which she could learn the rules and play with her father.

In this scenario, Olivia could benefit from the application by playing out multiple scenarios with her father – that would both inform her about secure web-development, as well as engaging her father with the world of cyber-security.



Figure 6: Royalty - Free Image [27]

Persona Two - Ben - Software Consultant

Ben is a software consultant for a medium-sized business. Although he has a degree in Computer Science, he is not an expert in cyber security.

As a consultant, Ben would like to develop software for his clients that is as secure as it is usable. Since this is done in a team, Ben would like to host an activity which brings together his co-workers (of varying experience) to participate in keeping up to date with current security trends.

His requirements are free, lightweight training material as he does not have the required funding, or time, to hire an expert security consultant.

To benefit his goal, the application will have multiplayer that will encourage his employees to challenge each-other in a stimulating way. Furthermore, the gamification aspect will not require his co-workers to work hard outside of their contracted hours.



Figure 7: Royalty - Free Image [28]

Persona Three - Samira - Security Analyst

Samira is a senior level security analyst whom completed her PhD on using gamification to teach security.

As an expert in her field, her passion for cyber security extends to all her colleagues, friends, and family. As such, she has a strong desire in raising awareness and eliminating the human error which lead to most security exploits.

Samira would benefit from the application by using it as a framework to map the mechanics to alternative cyber-risk scenarios, thus creating a training tool. Furthermore, she could use the application as a foundation to further her own research in gamification techniques.

3.2 Functional Requirements

The following requirements are necessary features that the application requires to reach a minimum viable product.

ID	Requirement	Description	MoSCoW
1	Multiplayer	Multiplayer turns between connected users	Must
2	Matchmaking	Users can host and join a game session	Must
3	Quit / Leave	Users can leave at any point	Must
4	Web Accessible	Playable on a web browser	Should
5	View Units	Users can see the status of all units	Should
6	View Description	Units and moves have an explanation	Must
7	View History	Users can see the game sequence	Must
8	Complete	Users can play a full version of the game	Must
9	Clear Goal	Gameplay loop and objectives are clear	Must
10	Single Player	Users can play single player	Won't
11	Movement	Units move with varied distances correctly	Must
12	Unit behaviour	Units have unique stats, moves and abilities	Must
13	Tutorial	Users can learn the rules and mechanics easily	Must
14	Gamification	Gamification mechanics should incite learning	Must
15	Challenge	The game should invoke critical thinking	Must

Table 3: Functional Requirements

3.3 Non-Functional Requirements

The following requirements outline how the application should perform.

ID	Requirement	Description	MoSCoW
16	Availability	Cross-platform playable (e.g. Tablet to Desktop)	Could
17	Ease of use	Easy to learn, understand and use	Should
18	Accessibility	Compliant with WCAG 2.1 and colour-blind friendly	Should
19	Secure	Playable on a secure website domain	Should
20	Correct	Factually correct in any cyber security concepts explored	Must
21	Performance	Gameplay should be 30 fps with low multiplayer latency	Must
22	Scalable	Scalable to support more than 2 players at any given time	Should
23	Learning	Users can learn deeper cyber security principles	Must
24	Extensible	Flexibility to add different scenarios, units, and abilities	Won't
25	Feedback	Provides feedback based on performance (e.g., Chess)	Could
26	Music	Good music that fits the theme and aesthetic	Should
27	Animations	Unit movement and abilities have satisfying animations	Should
28	Enjoyable	Depth of mechanics & dynamics that create meaningful play	Must

Table 4: Non-Functional Requirements

Is this font size okay, or stretching beyond the margins a little?

3.4 User Stories

The following user stories are derived from the functional, and non-functional, requirements (linked by the **ref** column). They illustrate the specific features the application should have in accordance to their MoSCoW priority.

ID	As a (role)	I want a/to.. (feature)	So that I can... (benefit)	MoSCoW	Ref	Difficulty
1	General User	View a unit's stats (HP/Defence)	Know who to attack	Must	5,6	Low
2	General User	View an ability's range, damage & cost	Know what I can do	Must	5,6	Low
3	General User	View a colour-blind friendly UI	Interact with the application easily	Should	18	Low
4	General User	View a unit's movement range	Where I can move to	Must	8, 11	Medium
5	General User	View a unit's ability attack range	Know who I can reach	Must	8, 11	Medium
6	General User	Move a unit	Attack a target	Must	8,12	High
7	General User	Attack a target	Kill an enemy unit	Must	8,9	Medium
8	General User	Disable the target	Prevent them from moving	Must	12	Medium
9	General User	Defend an ally unit	Prevent them from dying	Must	12	Medium
10	General User	Click on a move description	Learn more about the attack nature	Must	6	Low
11	General User	Click on a unit	Learn more about the attack vector	Must	7	Low
12	General User	Click on and view a history log	Follow the game's sequence of events	Should	7,17	Low
13	General User	Click on a unit map	Know how units relate to each other	Could	17	High
14	General User	View the units remaining	I know what units I can still move	Should	5	Low
15	General User	Pan the camera and/or zoom in	See the map from different angles	Could	18	Medium
16	General User	Unique animations for each ability	Differentiate abilities	Could	27	Very High
17	General User	Varied unity abilities	Enjoy varied gameplay	Could	12,24	High
18	General User	An intermediate tutorial	Quickly learn how to play	Should	13	Very High
19	General User	A help / status bar	Understand the gameplay	Should	13,23	Medium
20	General User	Multiplayer functionality	Host and join a session	Must	1,2,22	Very High
21	General User	Single player functionality	Play solo	Won't	10	N/A
22	General User	Quit / leave the session	Play with a friend	Must	3,8	Low
23	General User	Win or lose	Finish a game session	Must	8,9	Low
24	General User	Play on Desktop	Play on my PC	Must	16	Medium
25	General User	Play on Web Browser	Play on my browser	Should	4, 16	Medium
26	General User	Play on Tablet	Play on my tablet	Should	16	Medium
27	General User	Hear gameplay music	Become immersed within the game	Should	26,28	Medium
28	General User	Be able to play lag & bug free	Feasibly play the game	Must	21, 28	High
29	General User	Purchase & upgrade new units	Have a deeper reason to keep playing	Could	12,24	Very High
30	Employee	Receive feedback after a game	Know how to play better next time	Could	25	Very High
31	Employee	Play cross-platform sessions	Organise fun, training sessions	Could	1,2,16	High
32	Student	Learn about the OWASP #10	Understand attacks and mitigations	Must	20,23	Low
33	Researcher	Create new scenarios	So I can train & inform my friends	Should	14, 24	High

Table 5: Identifying User Stories

3.5 Constraints and Limitations

- The application must be optimised to run on a web browser with weaker graphics hardware (e.g. mobile devices)
- The WebGL 2.0 specification [29] has varied levels of support for different web browsers
- The application will require UI rescaling for different browsers; two aspect ratios of 16:9 and 4:3 will be prioritised
- The game must be fully playable with one button mapping to support touch screen gameplay
- Given the development time limit, only one game scenario will be created
- Due to Covid19, it will not be possible to evaluate a physical paper prototype of the application
- It will not be possible to iteratively improve the game's mechanics from sufficient playtesting

4 Design

This section provides a top-level overview of the physical design, game-sequence, and core components of the application.

4.1 Prototyping the Application

Figure 8 presents the initial wire-frame prototype of the application. Attack and Defence cards represent unit abilities which are explored in the following section.

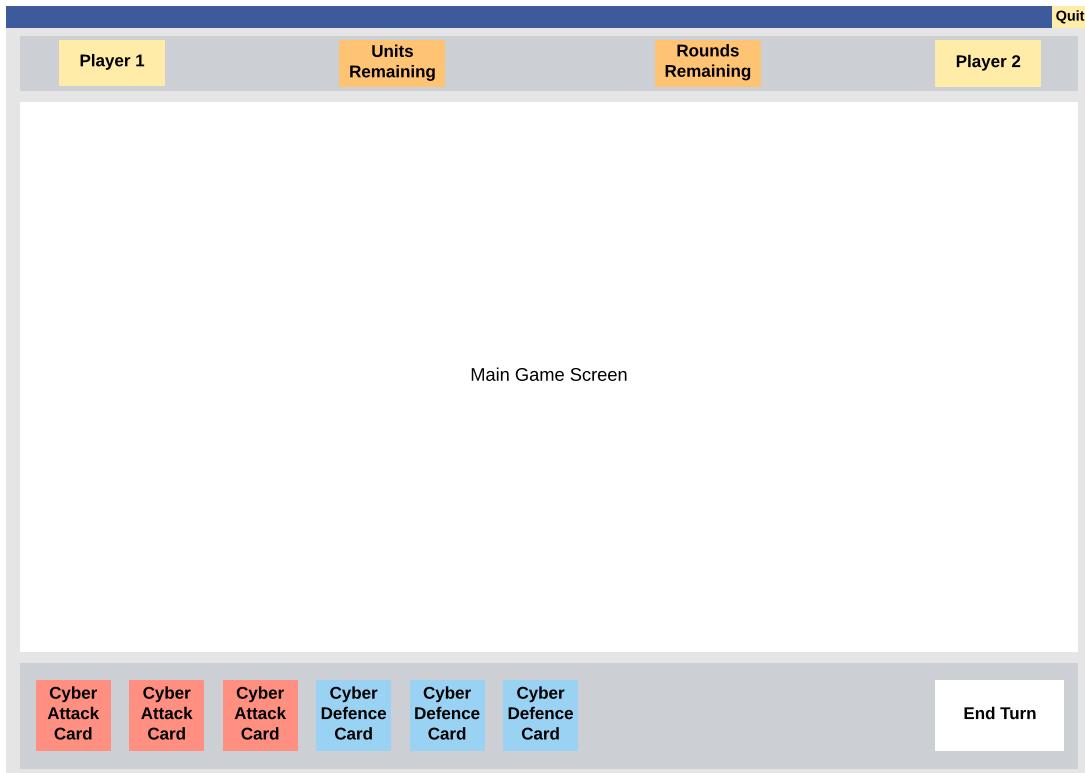


Figure 8: Wire-frame of Application

4.2 Applying the MDA Framework

The MDA (Mechanics, Dynamics, Aesthetics) framework [6] is the most well-known framework to identify the requirements of building a game. It is an iterative process in which the mechanics of the game are first outlined, and evaluated, to see what game-play loops (dynamics) emerge. Finally, the dynamics are mapped to a target aesthetic e.g., the scarcity of resources is an important dynamic for a survival title.

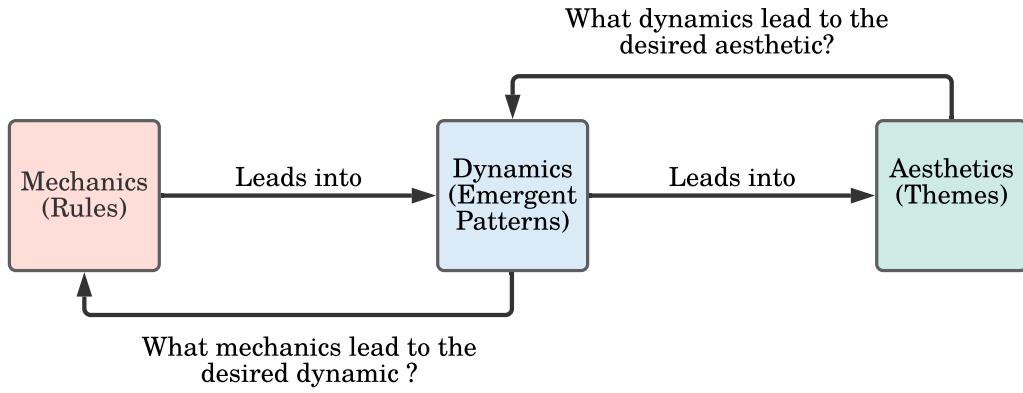


Figure 9: The Mechanics-Dynamics-Aesthetics Model [6]

4.2.1 Mechanics

The mechanics are the basic rules and actions a player can do within the game world [6].

They are as follows:

- Action Point (AP) represent a cost function for unit abilities
- Movement possibilities are blocked by enemy and friendly units
- Units can only attack once and move once per turn
- Units cannot move after attacking
- Units have a maximum defence which cannot be restored past this threshold
- Health points (HP) cannot be restored
- When a unit reaches 0 HP, it will be removed from the game world
- Each movable unit has 2 abilities, whereas the static units (database / web server) cannot directly be interacted with
- Abilities are limited by a distance range and can be blocked by other units
- Abilities can damage HP, defence, bypass shields, restore defence and disable other units
- Disabling a database / web server unit will halt AP gain for one turn
- Disabling movable units will prevent them from acting for one turn
- Abilities have unique effects versus certain units (e.g. SQL Injection deals critical damage to the database server)

4.2.2 Dynamics

The dynamics can be thought of as ‘the emergent behaviour that arises from gameplay when the mechanics are implemented’ [6].

They are as follows:

- Players may choose to target the web server/database to reduce AP gain
- Players may use shield destroying moves on heavy units
- Players may use cheaper moves or decide to save their AP
- Players may physically block units to protect them
- Players may use the bypass defence move to target units with a lot of defence but little HP (e.g. web server)
- Players may refer to the unit information to discover which moves have unique effects on certain units

4.2.3 Aesthetics

The aesthetics can be thought of as ‘the emotional response a game should illicit from the player’ [6].

They can be broken down into:

- Sensation (emotion-invoking)
- Fantasy (immersion)
- Narrative (story - rich)
- **Challenge (puzzles / obstacles)***
- **Fellowship (co-operative / multiplayer)***
- Discovery (open world)
- Expression (character creation)
- Submission (simulations)

* Where Fellowship and Challenge are the two key target Aesthetics for the purpose of this application.

4.3 Mapping the OWASP Top 10 to Game Mechanics

Figure 10 represents the choice of mapping security risks, exploits and mitigations to in-game mechanics. A description and justification of these mechanics are explored in section 5.8. The OWASP Top 10 was chosen as it represents a broad consensus of the most common web application security risks [30].

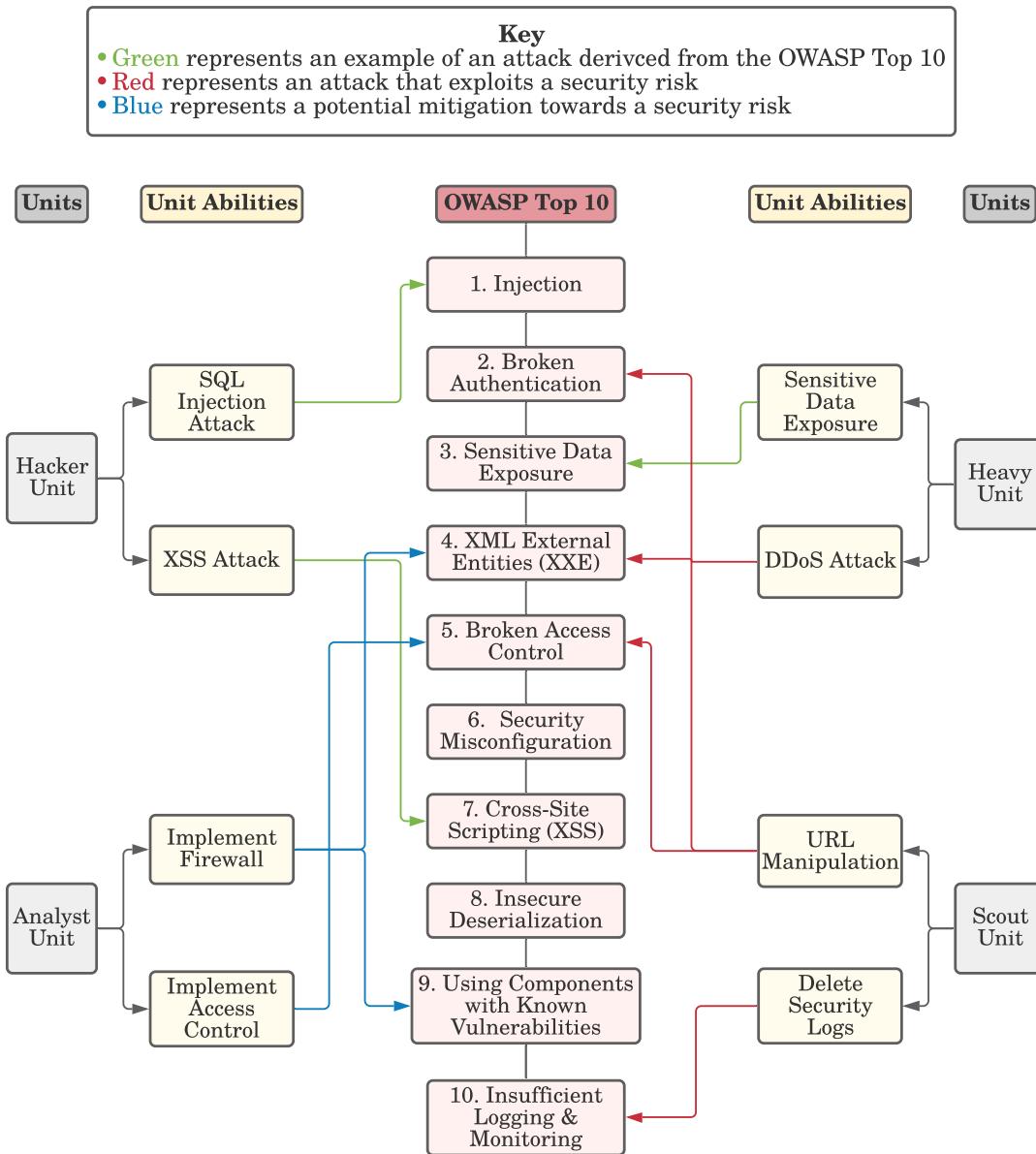


Figure 10: Mapping the OWASP Top 10 to Game - Mechanics

4.4 UML Modelling - A Class Diagram Representation

Figure 11 represents the core basis of functional code within the application but omits all testing, utility and helper scripts.

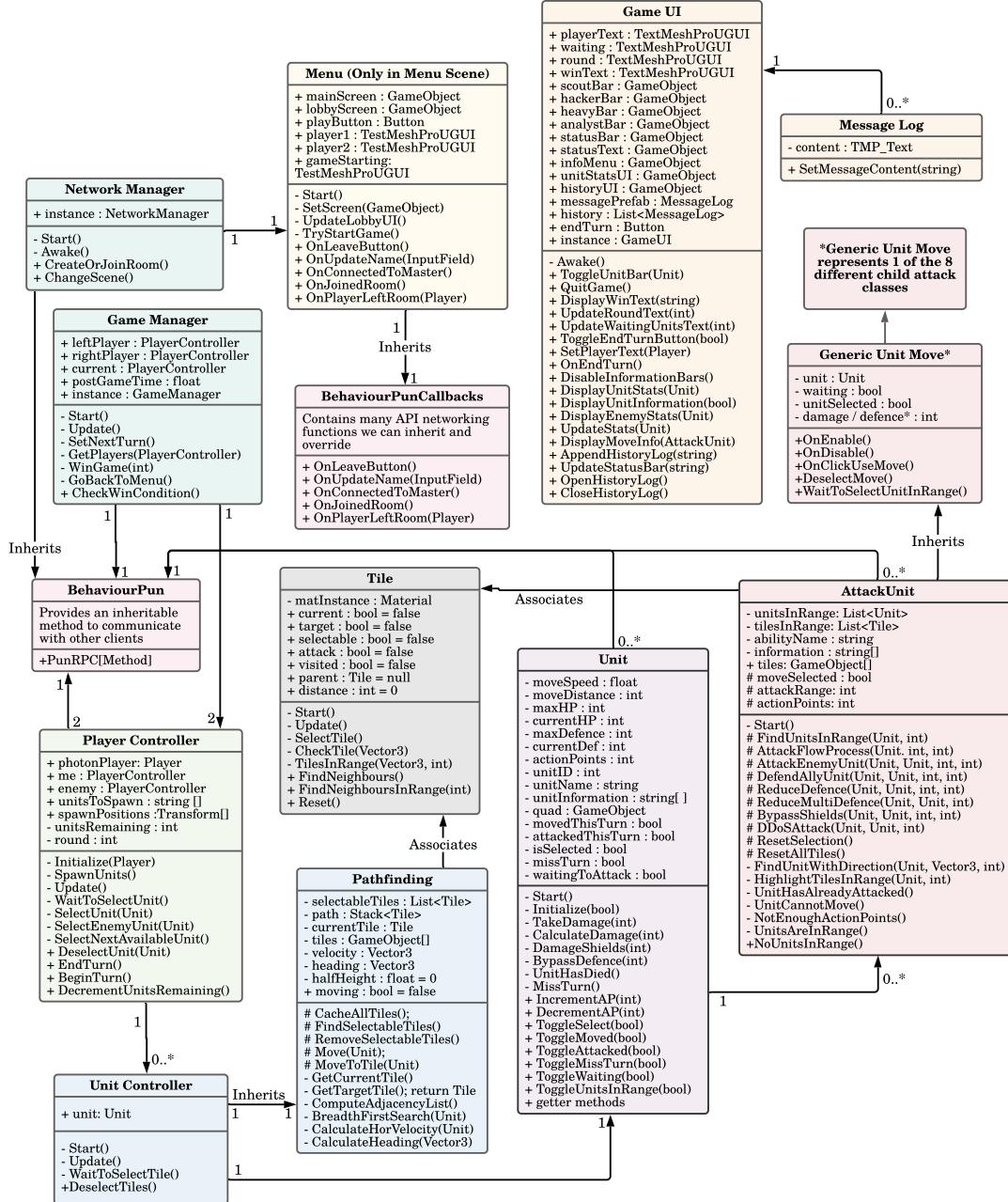


Figure 11: A Class Diagram Representation

4.5 Mapping the MDA Framework to Class Functions

Figure 12 highlights functions and components that specifically relate to the MDA framework. Due to the nature of Aesthetics being an emotional product of the game, in this diagram it represents all the physical assets and UI that the player interacts with.

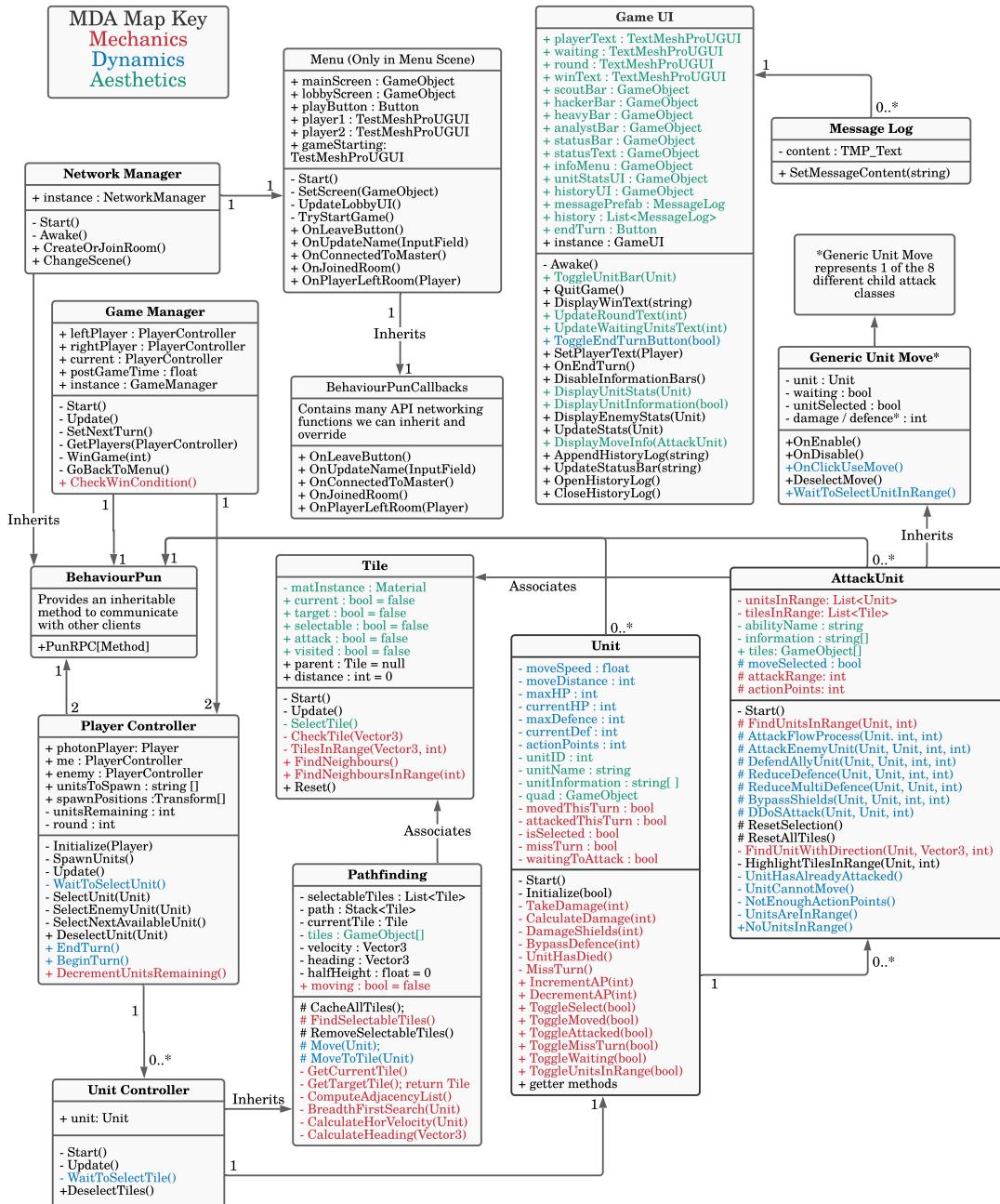


Figure 12: Mapping the MDA Framework to Class Functions

With reference to figures 11 and 12, an explanation of the core classes and components are as follows:

should i format this slightly difference, e.g. in a table or separate space between each class

Game Manager - a singleton instance of this class forms the backbone of the project as it controls the game's flow sequence, turn-manager, and game scene.

Behaviour PUN Callbacks - provides many networking functions that can be implemented to enable multiplayer functionality. This class is provided by the Photon Unity Networking 2 API.

Network Manager - a singleton instance of this class inherits the PUN2 API to handles hosting, joining, and leaving a room.

Sound Manager - a singleton instance of this class provides functions to invoke sound effects.

Menu - awaits user input (from within the game menu) to invoke networking functions within the network manager.

Game UI - handles the transition and display of the UI components.

Player Controller - listens for player input once per frame. It allows the player to select units and manages the players-turn and status of all units each turn. A singleton instance of this class is instantiated for each player.

Unit Controller - enables tile-selection and unit movement by inheriting functions provided from the Path-finding class. Each unit has an instance of this class.

Path-finding - utilises Breadth First Search to find the range of tiles that a selected unit can move to. It directly associates to each tile as it checks if it's unoccupied, and then highlights it green if it is a valid destination. The path-finding functions within this class were adapted from a tutorial available at Game Programming Academy [31].

Unit - contains unit information (e.g. health, defence, etc..) and functions relating to receiving damage, restoring defence and the unit's status (e.g. the unit has moved). Each unit has an instance of this class.

Attack Unit - provides functions for finding units in range, attacking, and defending units. It associates with the Tile class to accomplish this (e.g. is a unit located at this tile).

Generic Unit Move - represents the 8 unit abilities which are derived from the OWASP Top 10 (e.g. SQL Injection). These scripts inherit from Attack Unit for their shared functionality and are directly linked to the UI buttons which invoke the ability event upon user input.

4.6 UML Modelling - Activity Diagram

Figure 13 refers to the typical sequence of two players joining a game, matchmaking and then the taking turns until one player wins (by finishing all enemy units).

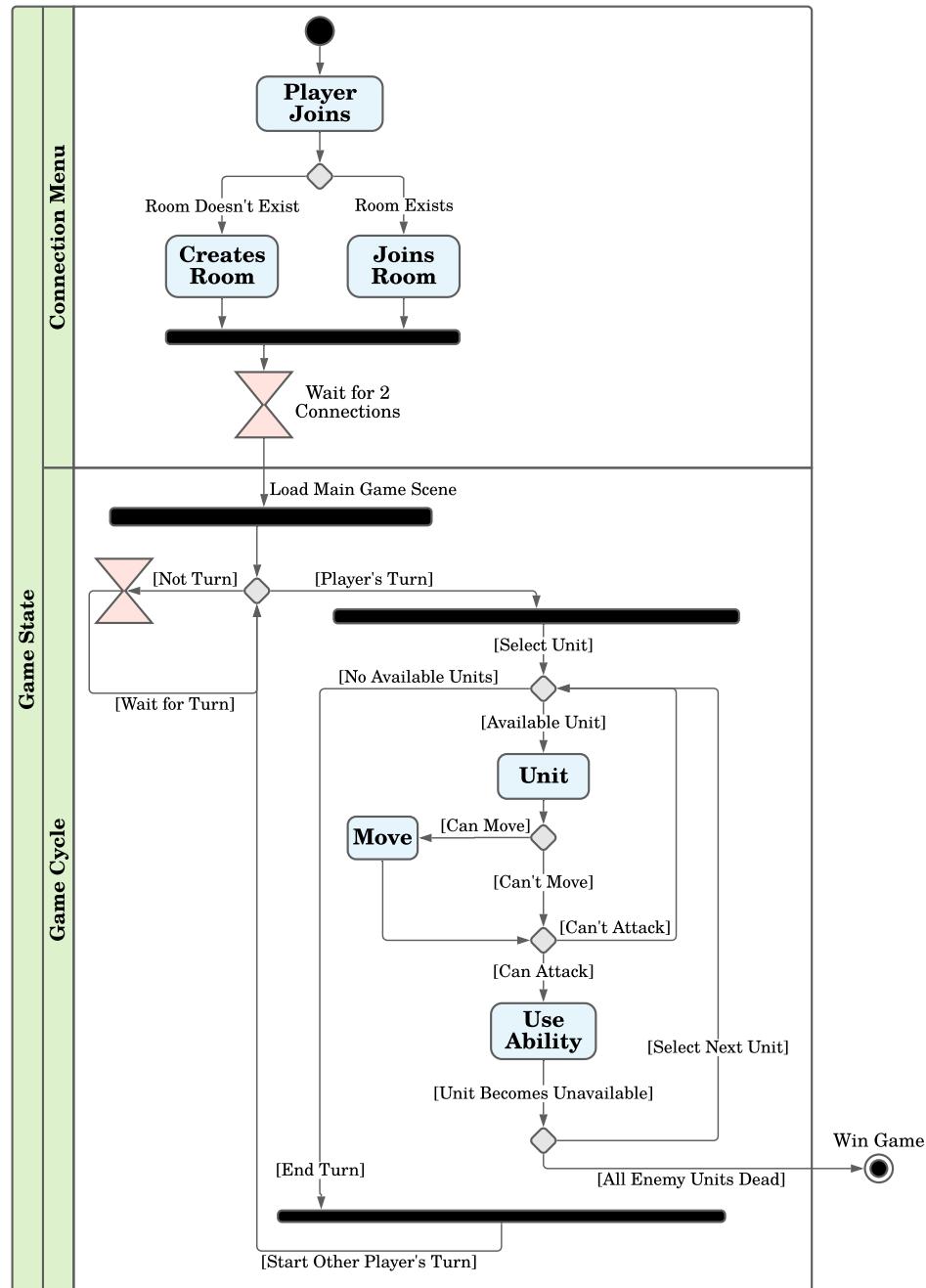


Figure 13: Activity Diagram of a Typical Match

4.7 Visual Design, Contrast and Accessibility

To provide the user with an immersive game-like experience, the UI and application needs to fit the theme of cyber security. As such, a colour palette was derived from NieR: Automata – a similarly themed title pivoted around machine intelligence. To ensure the application is visually suitable for the web, the following complimentary and harmonic colours were incorporated into the background [32].

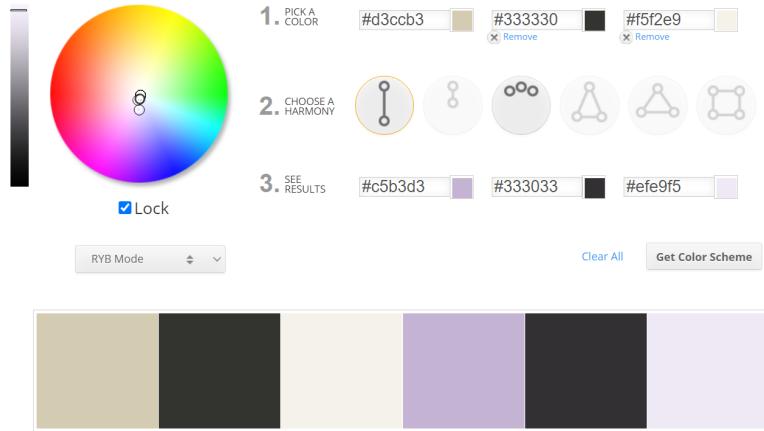


Figure 14: Identifying Complementary and Harmonic Colours

The Web Content Accessibility Guidelines (WCAG 2.1) requires that small text has a minimum contrast ratio of 7:1 [33]. All text used within the application has a contrast ratio of 18.75:1 and 13.05:1 which is compliant for all font-sizes.

Contrast Checker

[Home](#) > [Resources](#) > Contrast Checker

Foreground Color

 Lightness

Background Color

 Lightness

Contrast Ratio
18.75:1
[permalink](#)

(a) Larger Text [34]

Contrast Checker

[Home](#) > [Resources](#) > Contrast Checker

Foreground Color

 Lightness

Background Color

 Lightness

Contrast Ratio
13.05:1
[permalink](#)

(b) Smaller Text [35]

Figure 15: Contract Checking Text for Readability

4.8 Colour Blindness

Approximately 1 in 12 men (8%), and 1 in 200 women (0.5%), experience some form of Colour Vision Deficiency [36] which exemplifies the importance of designing with accessibility in mind.

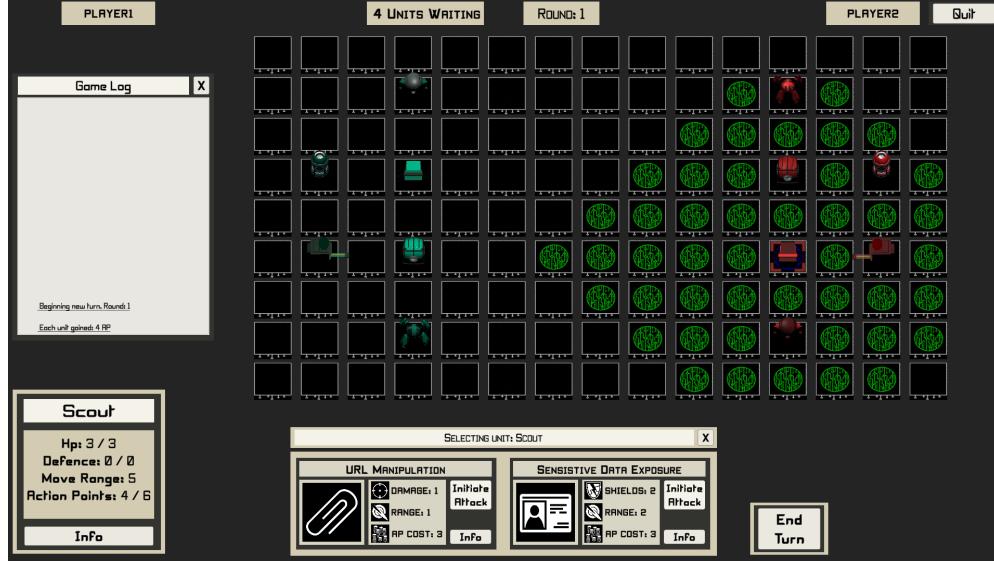


Figure 16: Initial Colour Scheme non-adhering to Accessibility Guidelines

Figure 16 presents the initial design and choice of unit-colour before amending these changes with colour blindness in mind. As such, a colour-blind friendly palette [37], was implemented, alongside a royalty-free background pattern [38], which better contrasts the game-scene.



Figure 17: Revised Colour Scheme for the Desktop Application

4.9 Designing for the Web and Mobile

Due to technical limitations of rendering the tile design in WebGL, an alternative simpler design was developed for WebGL builds (see figure 18).

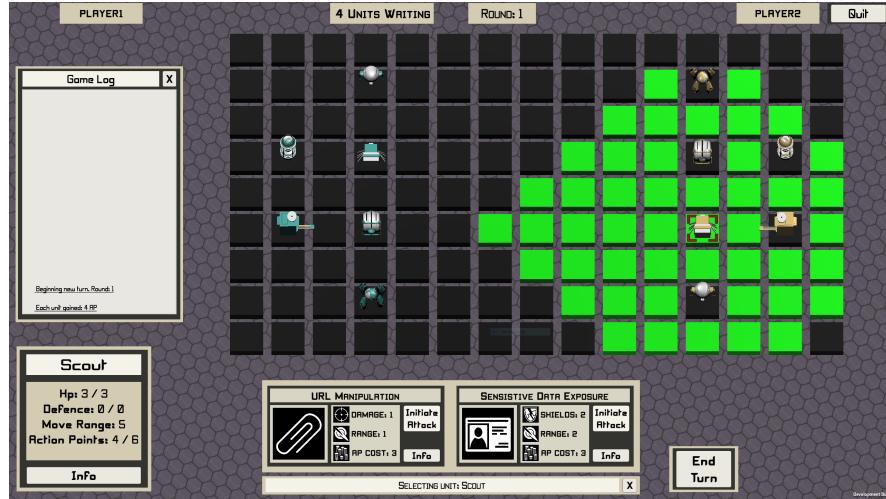


Figure 18: WebGL Build (16:9 Aspect Ratio)

To fulfil the requirements of developing for tablet devices, the applications UI supports the standard monitor aspect ratio of 16:9 (and scales to ultra-wide 21:9), as well as the standard tablet aspect ratio of 4:3. All controls support touch screen with a typical point-and-click button mapping.



Figure 19: WebGL Mobile Build (4:3 Aspect Ratio)

4.10 Design Patterns for Unity & C#

The following section exemplifies the recommended design patterns for Unity.

4.10.1 Component Design Pattern

This is the most common design pattern as Unity is a component-driven engine, in which game objects, scripts and functions are typically broken down into smaller sub-components [39].

The Unit game object is a good example of this as, all unit functionality (stats, movement control, networking operations and rigid bodies / mesh colliders), are broken down into sub-components and stitched together - figure 20. Similarly, to object-orientated programming, this approach enables components to be decoupled, re-arranged, and re-used easily.

A benefit of this approach is that unit's stats and abilities assigned (via the UI) can be modified and re-arranged at any time, allowing for flexibility of using one unit as a model for creating completely new units.

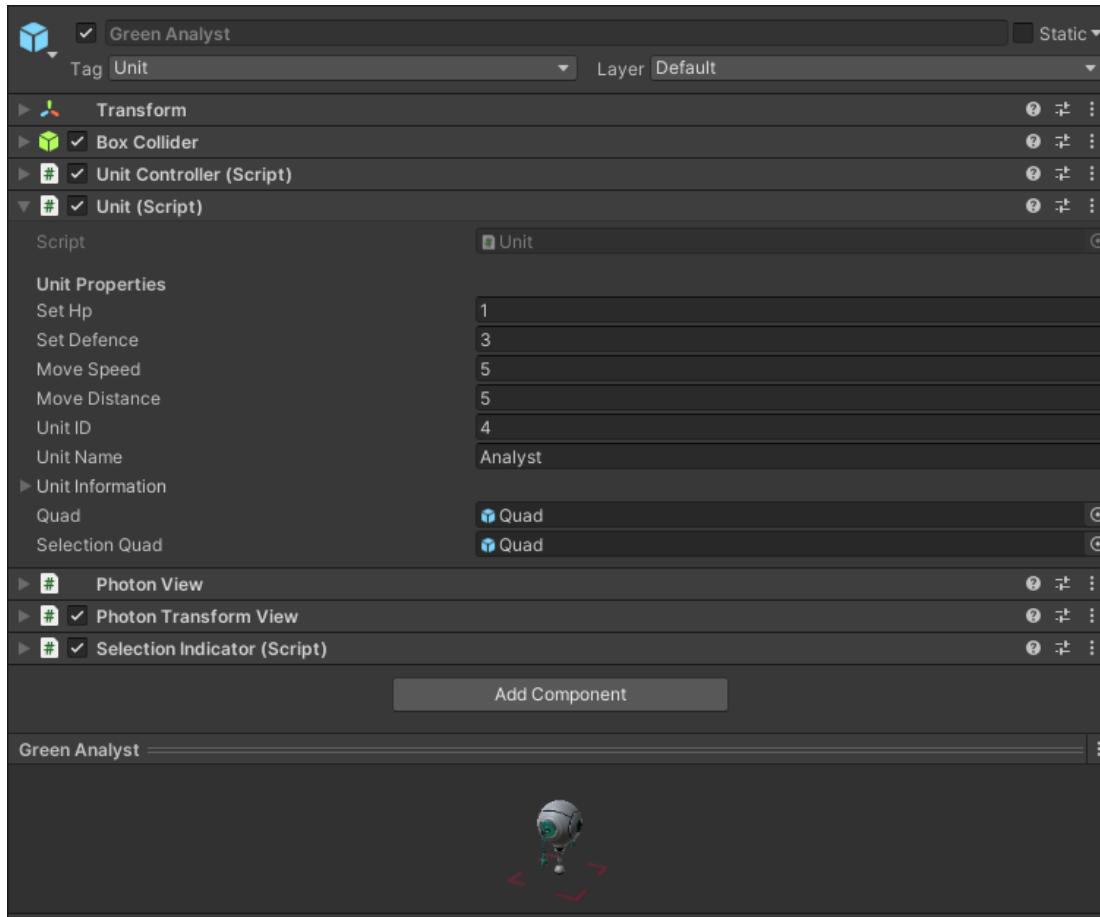


Figure 20: Example of Unit Components

4.10.2 Singleton Design Pattern

The singleton pattern involves creating a public static reference (of a non-static class) such that only one instance of that class exists [39]. This is incredibly useful for managers (e.g. the Game Manager) as it is not possible to instantiate static classes. However, having a single, static reference is versatile in that it stores important information about the state of the game – and can be publicly invoked by all other components without needing to worry of duplicated instances.

```
public class GameManager : MonoBehaviour {

    // Static singleton instance of the Game Manager
    public static GameManager instance;

    // Assign instance upon initialisation
    private void Awake() => instance = this;
}
```

Figure 21: Singleton Pattern

4.10.3 Flyweight Design Pattern

The flyweight pattern is designed to save memory by re-using instances of the same object [39]. For example, each time a tile changes colour, if a material instance is not saved and re-used, a new material object would be created which leads to unnecessary memory losses. This is important because as the size of the game scales (with hundreds of tiles), lag spikes will occur more frequently as the garbage collector must be invoked to remove old material instances.

```
private Material _matInstance;

// Cache material component on initialisation
private void Start() {
    _matInstance = GetComponent<Renderer>().material;
    _matInstance.color = Color.black;
}
// Re-use material instance each time the tile is updated
public void MarkSelectable() {
    selectable = true;
    _matInstance.color = Color.green;
}
```

Figure 22: Applying the Flyweight Pattern to Tile's Changing Material

5 Implementation

Unity [40] was chosen as the development environment because one of the key project requirements was to design a game that could be run on multiple platforms, including the web and mobile devices. Unity provides functionality for exporting to a vast range of systems - including WebGL [41].

Phaser3 [42] was considered as an alternative, web-development based solution. However, Phaser3 is a relatively small open source project whereas Unity has the largest online community, ample resources and integration for supporting frameworks.

5.1 Multiplayer with Photon Cloud

Photon Cloud provides a software as a service (SaaS) solution to host cloud servers for free up until 20 concurrent players [43]. Beyond this, Photon Cloud offers paid services for more scalable solutions.

Photon Cloud works with an API - Photon Unity Networking 2 (PUN2) [44], which enable developers to integrate multiplayer functionality into their games. PUN2 was chosen as it has strong integration with Unity and even supports cross-platform networking which fulfils requirement 16.

Mirror [45] is an alternative open source library which was built to replace Unity's previous deprecated multiplayer libraries (UNET). Although Mirror is completely free to use, relatively simple and highly covered, Mirror does not provide any functionality or cloud services for externally hosting a game server. As this would require running a server constantly, this is would not be a viable long-term solution.

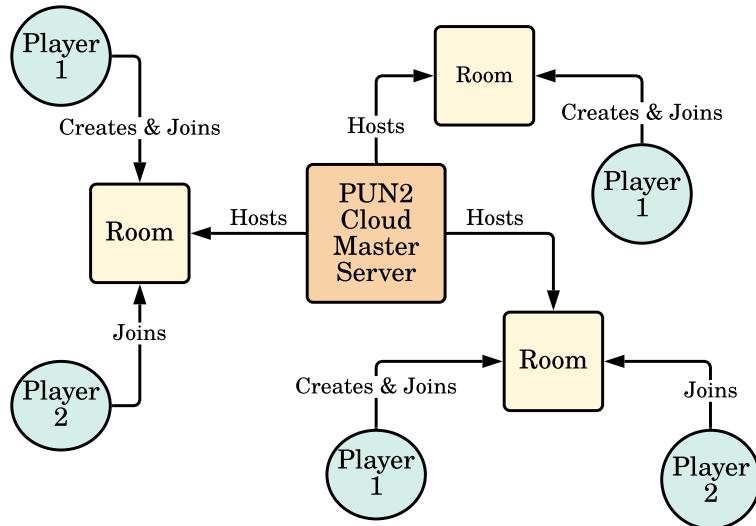


Figure 23: PUN2 Architectural Design

5.2 Photon Unity Networking 2 Architecture

PUN2 follows a classic Server - Client architecture (see Figure 23). When a client connects to the Photon network, it invokes the ‘JoinOrCreateRoom()’ function which connects the client to an non-full room (if it exists), or creates one if there are no rooms available.

The first client that creates a room becomes the master client which is responsible for hosting the game. If the master client disconnects for any reason, there are options for the next available client (if present) to be promoted to a new master client. However, since this project is only designed for two players, it returns the other player to the main menu and ends the room session.

5.3 Remote Procedure Calls - RPCs

PUN RPCs [46], provided by the PUN2 API, enable client - client communication in which programmed functions can be invoked on the opponent’s client remotely. As both clients are running different, independent images of the game, RPCs are necessary to maintain concurrency by matching these images and state variables. For example, after a unit has attacked another unit, it will remotely call the “Take Damage” function which will invoke the function on the recipient client.

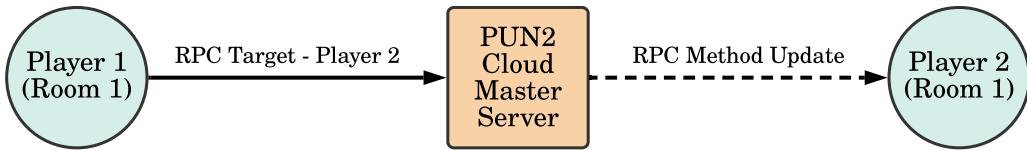


Figure 24: RPC Communication to the Other Client

By calling ‘RPC Target.All’ however, the function is executed on all clients connected in the room (including the place of origin). This is the most often case since the majority of multiplayer functions are state-changing (such as beginning a new turn, and editing a units variables), and these values must remain concurrent across both users.

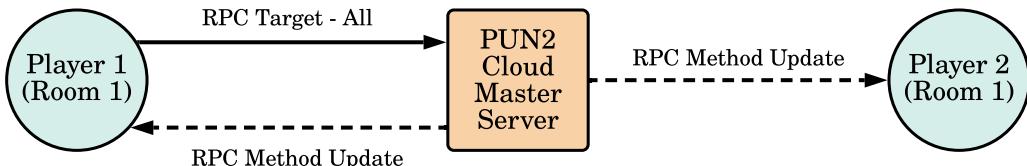


Figure 25: RPC Communication to All Users

5.4 Opening the Application

Figures 26 & 27 present the welcoming scene which facilitates auto matchmaking (for simplicity). In an earlier iteration, a lobby search system was created for 4-player games, however this was reduced as it's not necessary for 2-player games.

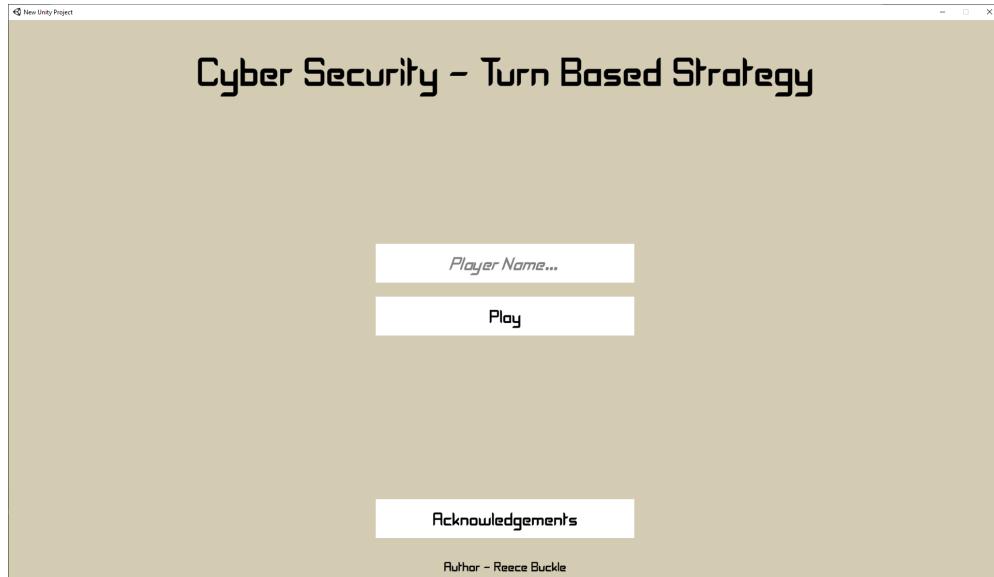


Figure 26: Welcome Menu

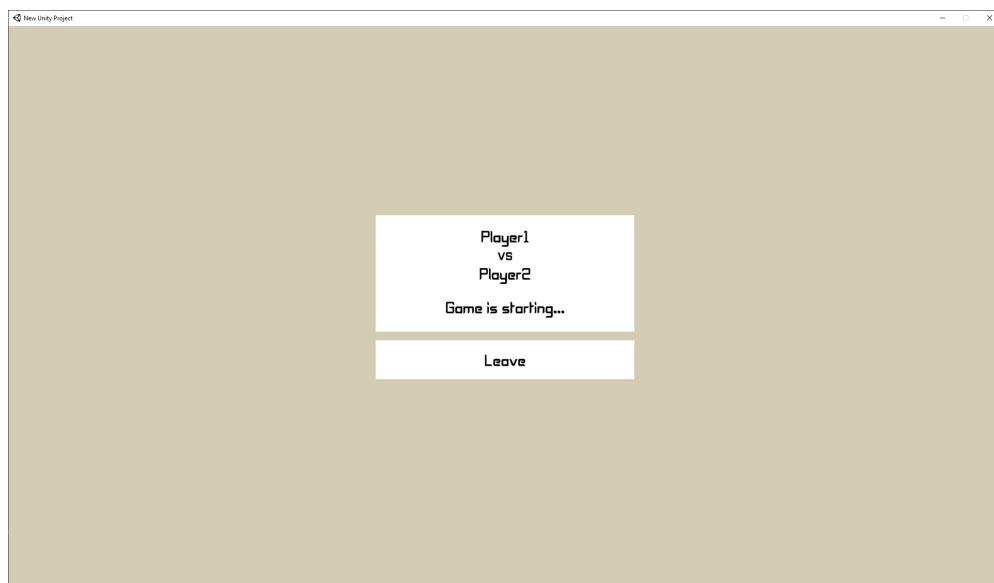


Figure 27: Connection Screen

5.5 Implementation of Unit Movement

A unit's movement distance parameter can be freely changed for any unit by setting its movement range within the inspector. Figures 28 & 29 demonstrate the range of tiles a selected unit can move to. Note these are blocked by other units.

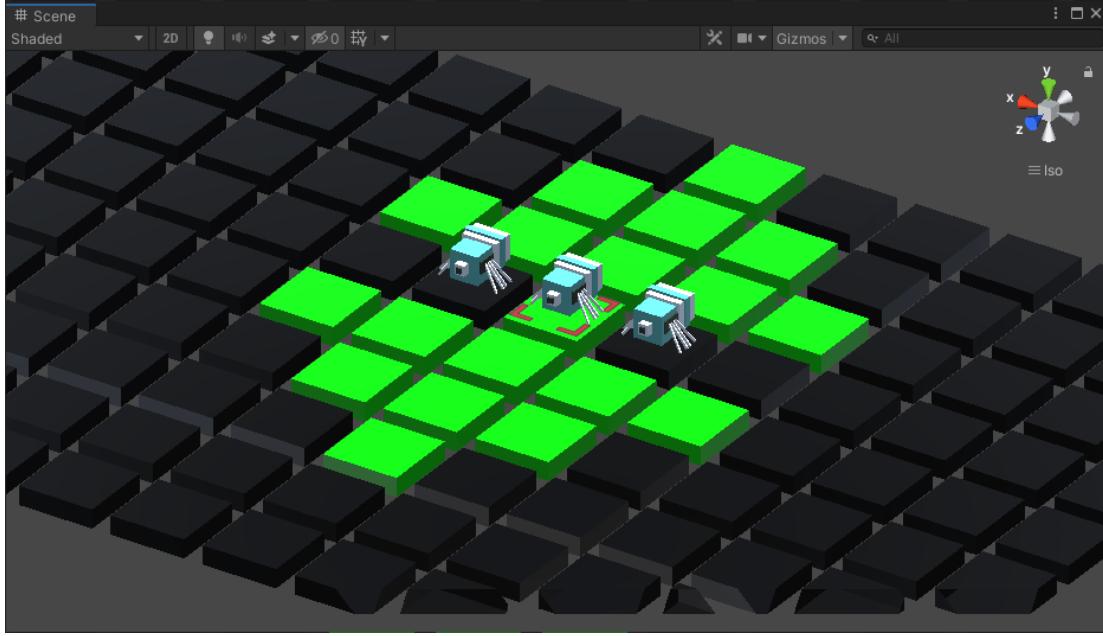


Figure 28: Movement Options for a Test Unit with Range: 3

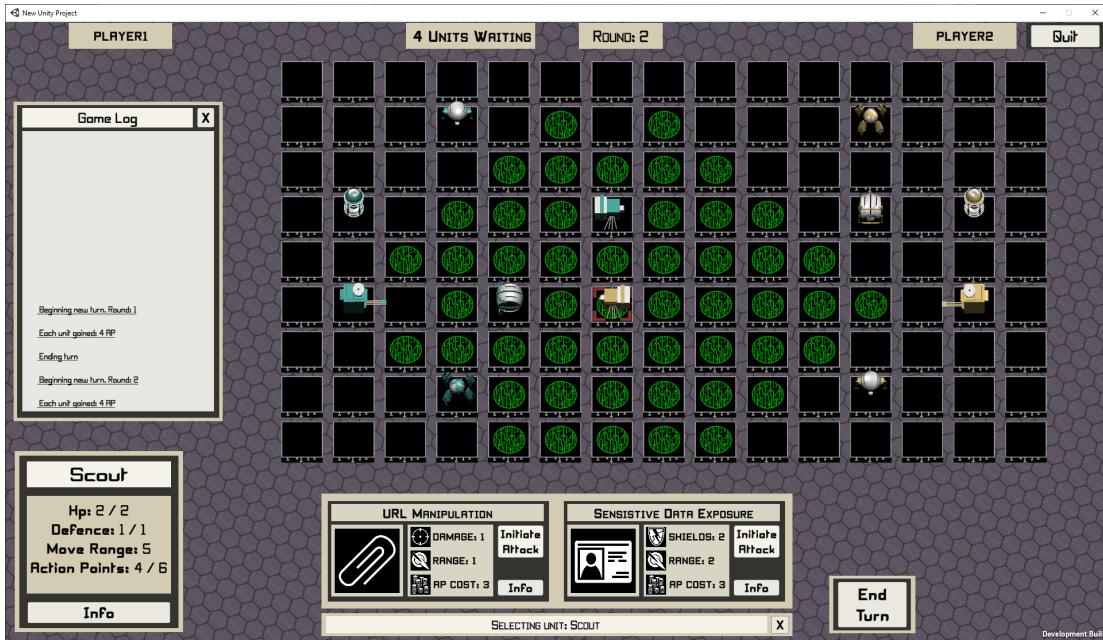


Figure 29: Movement Options for a Scout with Range: 5

5.6 Movement Mechanics - Breadth First Search

To calculate the selectable tiles in range of a unit, Breadth First Search (BFS) was implemented as it always returns the shortest path to the destination.

When a unit is selected, a ray-cast is sent below to find the tile underneath. From this tile, four more ray-casts are emitted north, east, south, and west to discover any neighbours. If these neighbours are walkable, they are added to adjacency list attached to that tile. Occupied tiles are ignored.

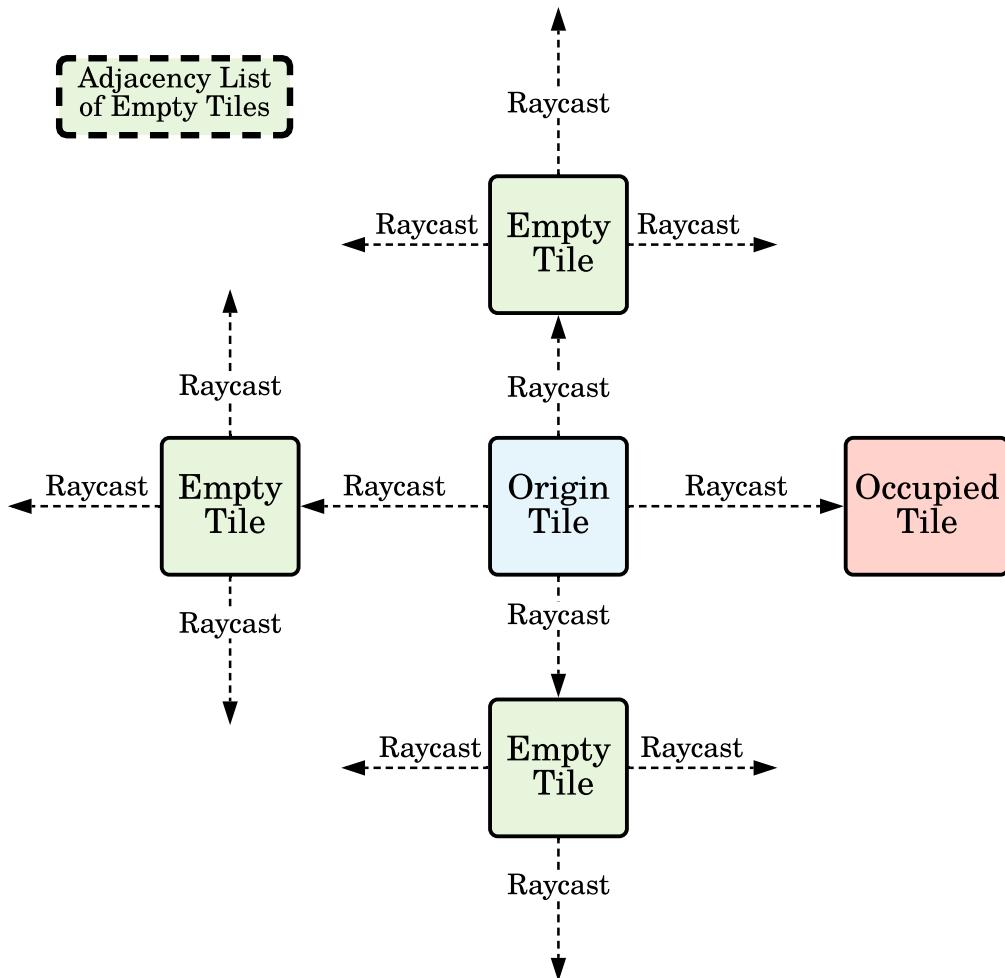


Figure 30: Ray-casting to Adjacent Tiles

BFS utilises a queue data structure. The starting tile is enqueued first. This is then dequeued such that it is marked as selectable, and all adjacent (walkable) tiles can be processed. Each tile is marked as visited so that these tiles are not checked twice. See Figure 31 for a diagrammatic representation, and figure 33 for a programming explanation.

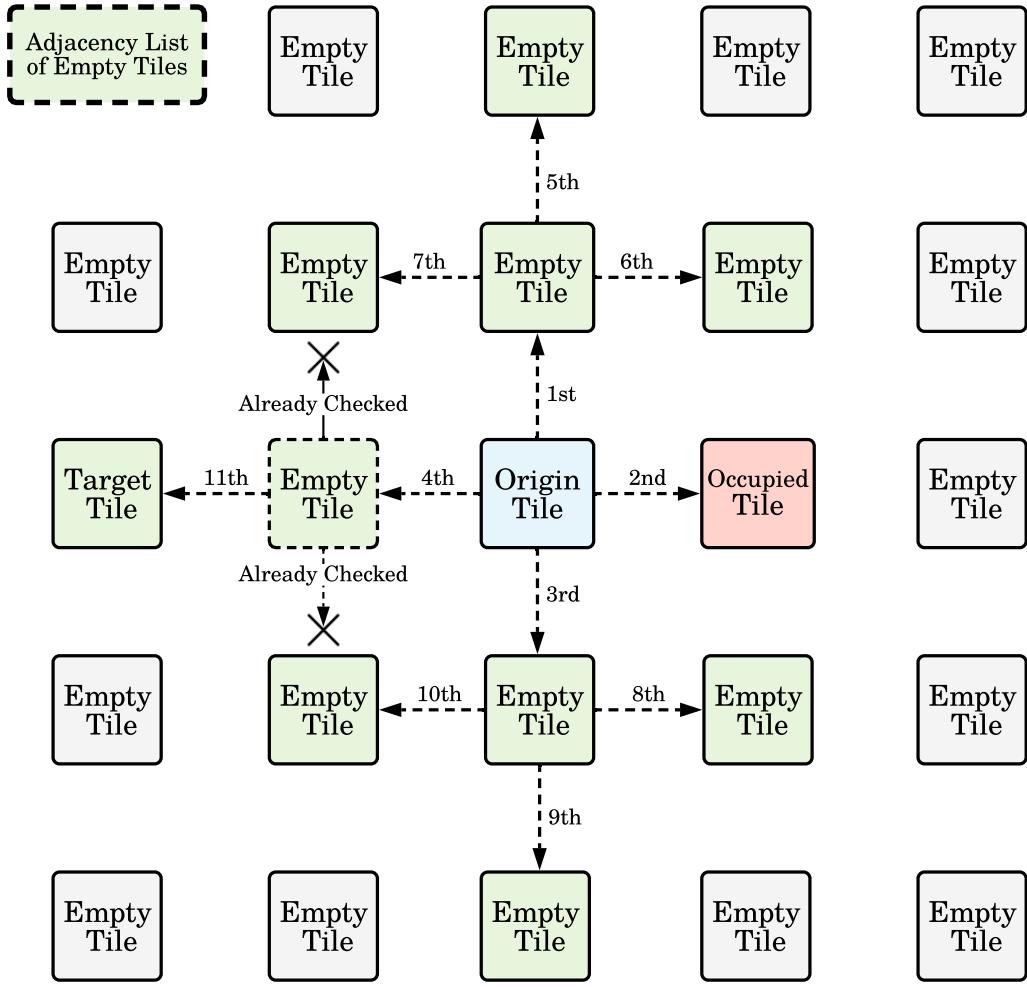


Figure 31: Finding Selectable Tiles via Breadth First Search

For each adjacent tile that is processed, its parent tile is assigned to it. When a target tile is selected, its parent is recursively accessed until we arrive back at the origin tile (see Figure 32). As such, this always returns the shortest path to the destination.

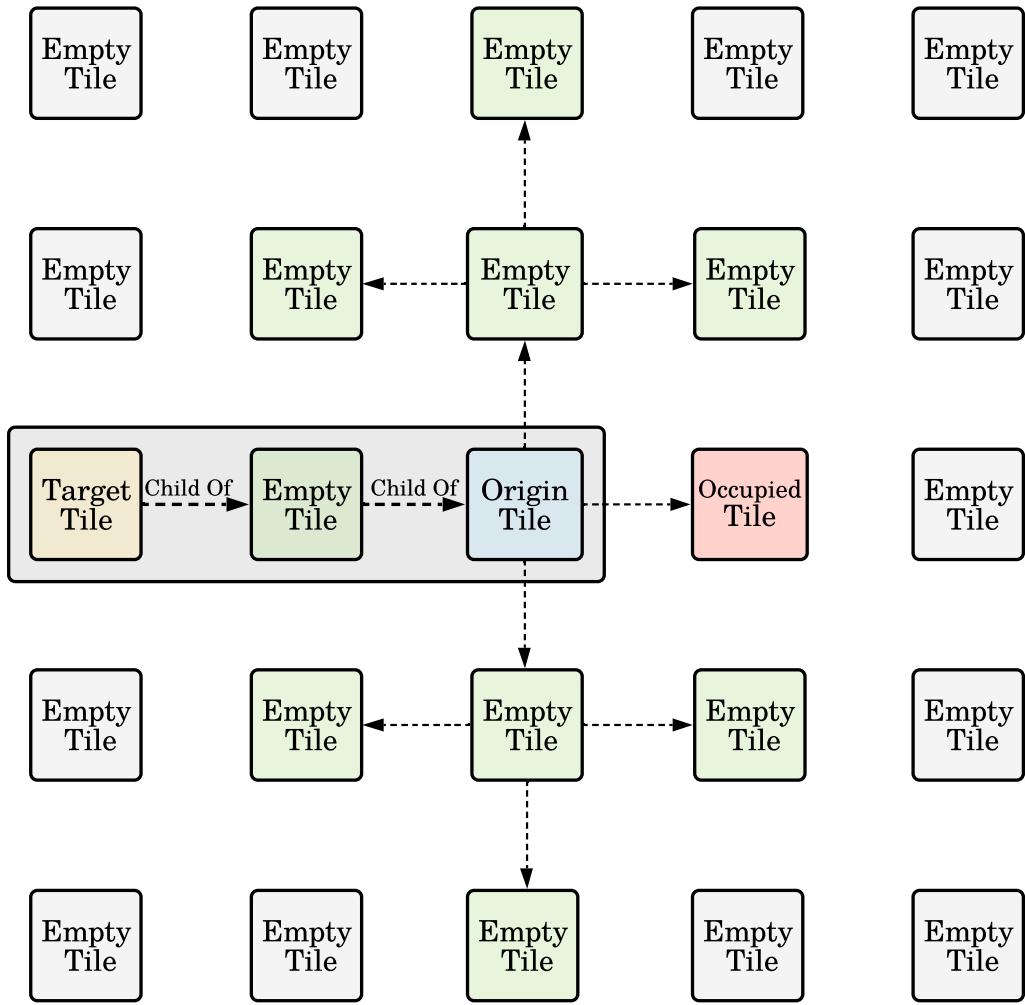


Figure 32: Finding the Shortest Path to Target Tile

Figure 33 describes the core portion of breadth first search implemented in this manor.

```
private void BreadthFirstSearch(Unit unit) {  
  
    Queue<Tile> tiles = new Queue<Tile>();  
  
    // Enqueue the tile underneath selected unit  
    tiles.Enqueue(currentTile);  
  
    // Mark as visited so this tile isn't processed twice  
    currentTile.visited = true;  
  
    // Begin dequeuing whilst enqueueing adjacent tiles  
    while (tiles.Count > 0) {  
  
        // Dequeue the first tile and mark as selectable  
        Tile tile = tiles.Dequeue();  
        selectableTiles.Add(tile);  
        // Highlight the tile green  
        tile.MarkSelectable();  
  
        // If the tile is outside of range, continue to  
        // the next tile in the queue  
        if (tile.distance >= unit.GetMovementDistance())  
            continue;  
  
        // Search through the 4 adjacent tiles  
        foreach (Tile t in tile.adjacencyList) {  
  
            // If tile has been visited, continue to the  
            // next tile in the adjacency list  
            if (t.visited) continue;  
            // Assign the parent of the adjacent tile  
            t.parent = tile;  
            t.visited = true;  
            // Update distance of tile from unit to keep  
            // track of how far we are from the starting  
            // tile  
            t.distance = 1 + tile.distance;  
            // Enqueue tile from adjacency list  
            tiles.Enqueue(t);  
        }  
    }  
}
```

Figure 33: Implementing Breadth First Search

5.7 Ability Range

Calculating an ability's range also utilises ray-casts to find units and tiles in range. However, this does not utilise BFS as it works in a line-of-sight manor. When a unit is selected, a target selection box is displayed underneath for a visual cue. Values for range, damage and action point cost can be set freely within the inspector.

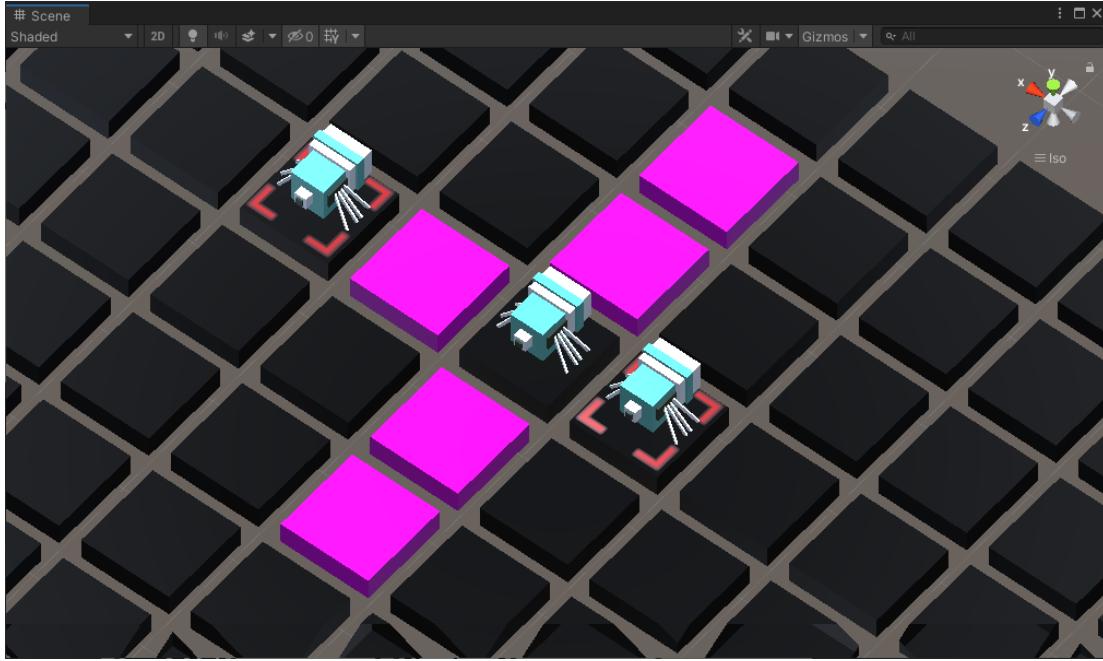


Figure 34: Demonstration of a Test Ability with 2 Range

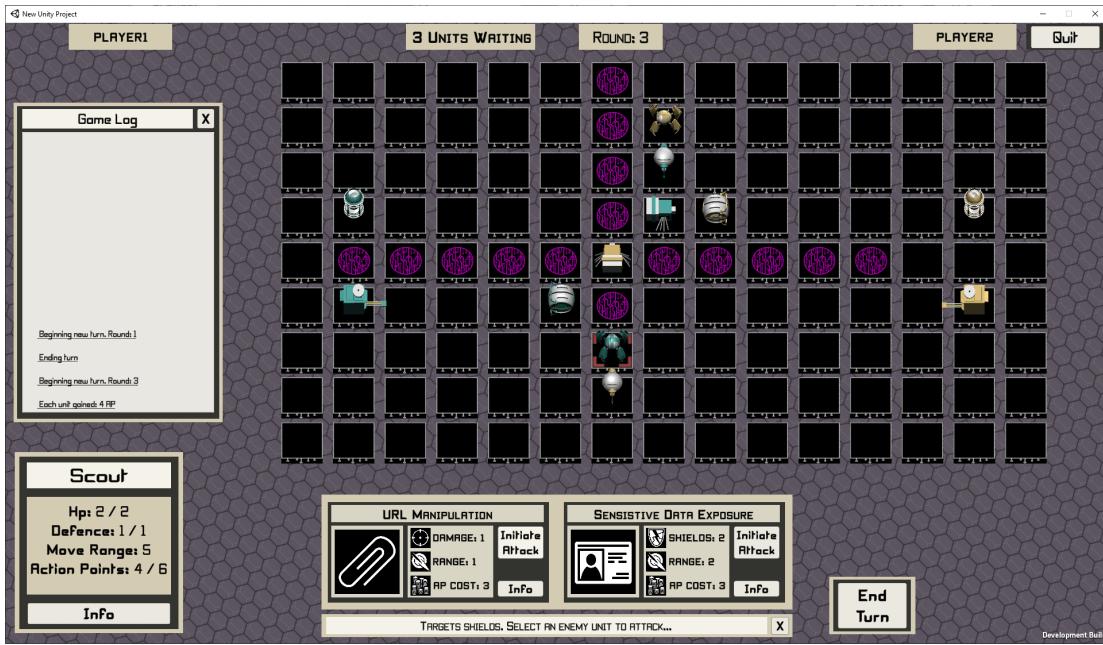


Figure 35: An Ability with 5 Range

5.8 Implementation of Units and Unit Abilities

The following sections describe the implementation of selectable units and their abilities which are derived from the OWASP Top 10 (2017) [30]. The descriptions of each ability and unit are loosely adapted from the description given in the application. All model assets are royalty - free and accessible on Blend-swap.

5.8.1 Implementation of Scout Unit

The Scout unit represents exploiting broken control and exposing sensitive information from poorly designed web applications. It has access to the following moves: URL Manipulation and Sensitive Data Exposure (see Figure 37).

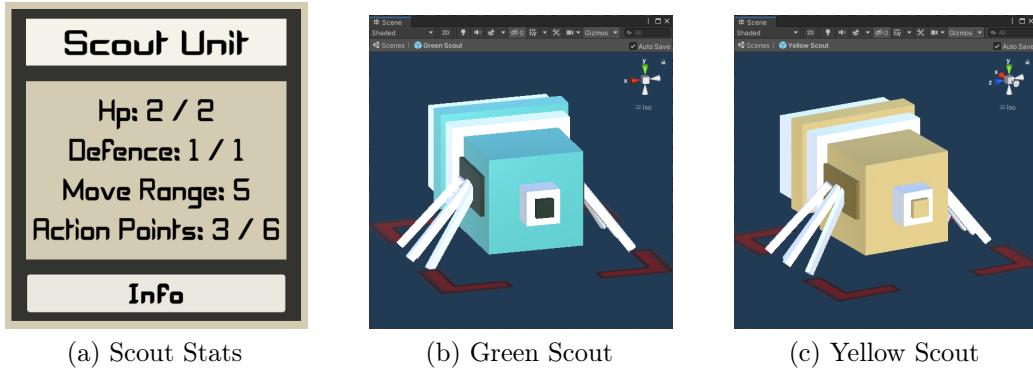


Figure 36: Scout Unit Stats and Model [47]

As **URL Manipulation** typically involves changing the URL parameters in order to bypass authorisation control, the core ability of this move is to bypass the enemy unit's defence in damage calculations. This makes it a good move against units with high defence, but low HP (such as the front-end web server). This move is close range as it requires the unit to directly interact with the web application.

Sensitive Data Exposure includes leaking personal, account and financial information [48]. Improper handling of sensitive data in transit or storage lead to data breaches which can cripple a company's reputation. In more severe outcomes, such information could be used to commit fraud and identity theft. As such, this move is a highly damaging defence-reducing attack.



Figure 37: Scout Abilities - URL Manipulation and Sensitive Data Exposure

5.8.2 Implementation of Hacker Unit

The Hacker unit specialises with direct attack vectors typical of penetration testers who can prod a web application in search of vulnerabilities. SQL injection and XSS scripting are the two most common website security attacks, with XSS attacks occupying approximately 40% of all cyber-attacks [49].

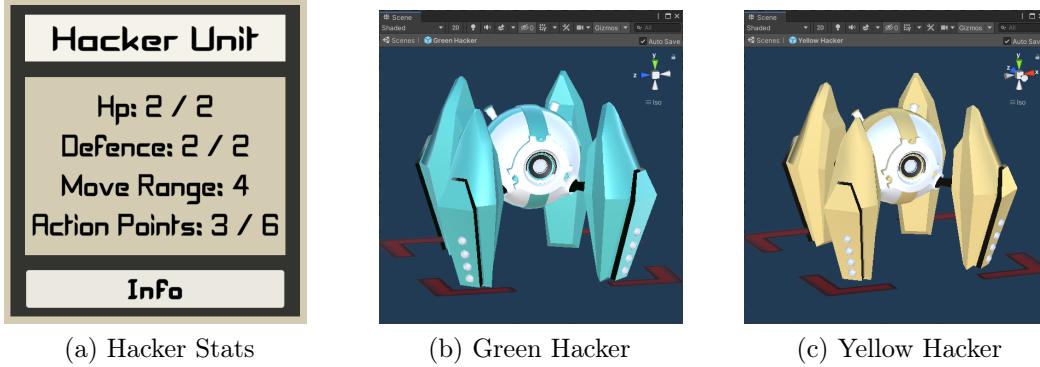


Figure 38: Hacker Unit Stats and Model [50]

Cross-Site Scripting (XSS) attacks can occur when a web application fails to validate user input by escaping special characters. As a result, an attacker can execute scripts in a victim's browser, hijack their session data and even redirect them to other phishing sites [30]. Therefore, this move is a highly damaging close range move which does +1 critical damage to the web server.

Sequel Query Language (SQL) injection exploits the interactivity between the web application and commands that query the back-end database. Using SQL injection, it is possible to bypass authorisation, extract sensitive information from databases and even manipulate database entries with your own custom information [48]. Therefore, SQL Injection is a longer range move which does +1 critical damage to the database server.



Figure 39: Hacker Abilities - SQL Injection and Cross-Site Scripting Attack

5.8.3 Implementation of Heavy Unit

The Heavy unit represents physical disruption by disabling operations externally and internally. It has access to the following moves: Distributed Denial of Service and Delete Security Logs (figure 41).

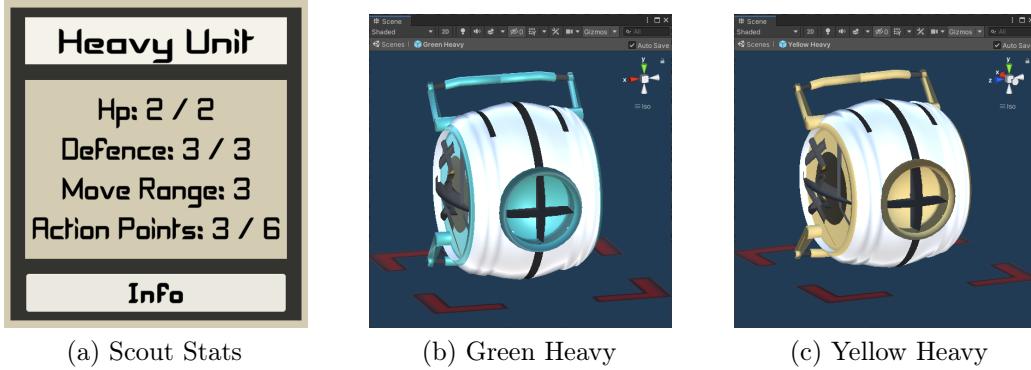


Figure 40: Heavy Unit Stats and Model [51]

A **Distributed Denial of Service (DDoS)** involves using a network of devices in order to flood a web application, network or server with overwhelming internet traffic - and disable it from being able to handle requests [49]. As such, this move will disable any movable unit for 1 turn. If used on the web/database server, it will halt action point for all units for one turn.

Delete Security Logs exploits Insufficient Logging & Monitoring [30]. Insufficient logging & monitoring (e.g recording anomalies, failed login attempts etc) allows an attacker to persist in a compromised system without being detected [48]. As this can affect the entire system, this move is a multi-targeting move that lowers the defence of all units in range.

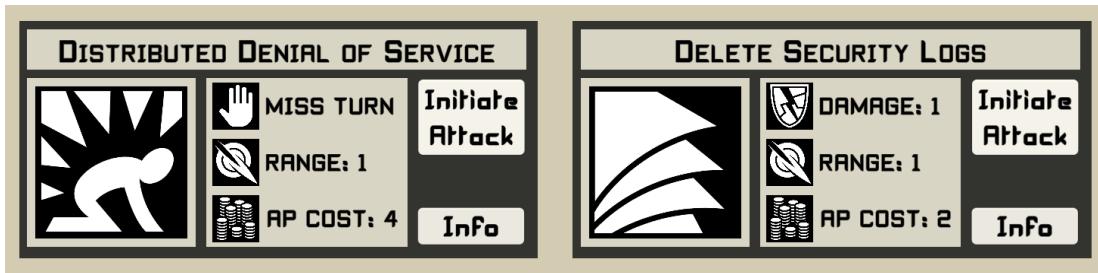


Figure 41: Heavy Abilities - DDoS Attack and Delete Security Logs

5.8.4 Implementation of Analyst Unit

The analyst is designed to represent the role of security analysts who specialise in defending a network. This includes implementing security firewalls, keeping systems up to date and monitoring & logging application activity.

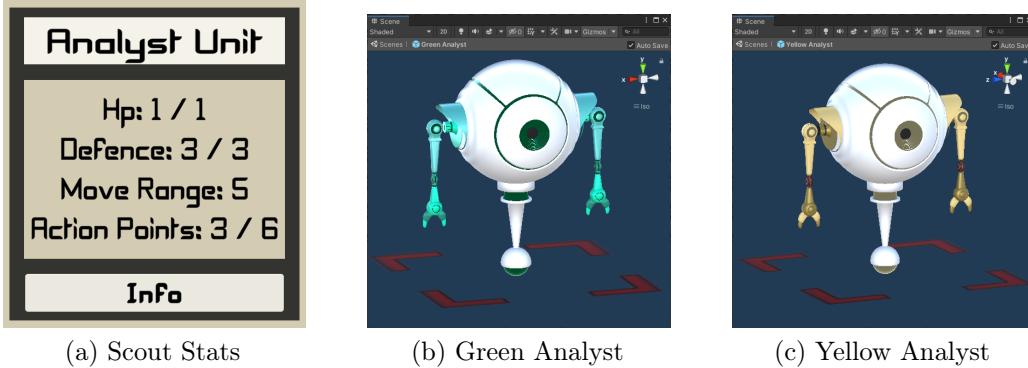


Figure 42: Analyst Unit Stats and Model [52]

A **Firewall** is required to filter HTTP traffic and keep web-application components safely isolated. A web application firewall (WAF) utilises rules to mitigate common attacks such as XSS and SQL Injection [53]. As such, this move restores the shields of one unit in range.

Implement Access Control is designed to mitigate Broken Access Control [30]. A good access control system should be implemented once, and periodically reviewed. Control policies based on user roles should enforce the least amount of privileges each user requires to complete their interaction within the web application [54]. As such, this move is multi-targeting and restores the defence of all units range.

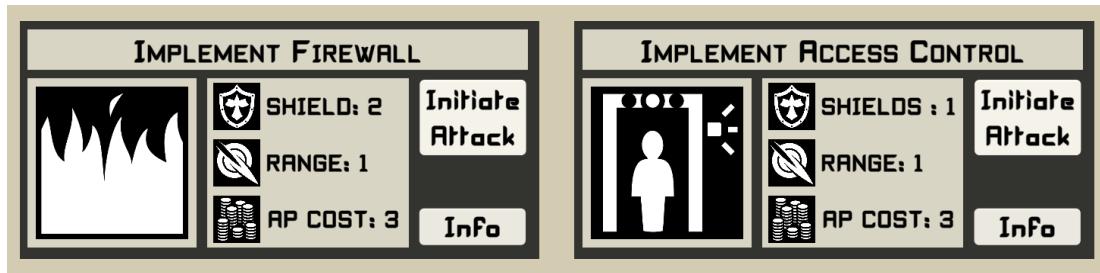


Figure 43: Analyst Abilities - Implement Firewall and Access Control

5.8.5 Implementation of Server Units

The web server is responsible for displaying web pages and handling HTTP network requests from all users accessing the web application. As such, it needs to be able to handle all user requests correctly whilst being robust to security risks.

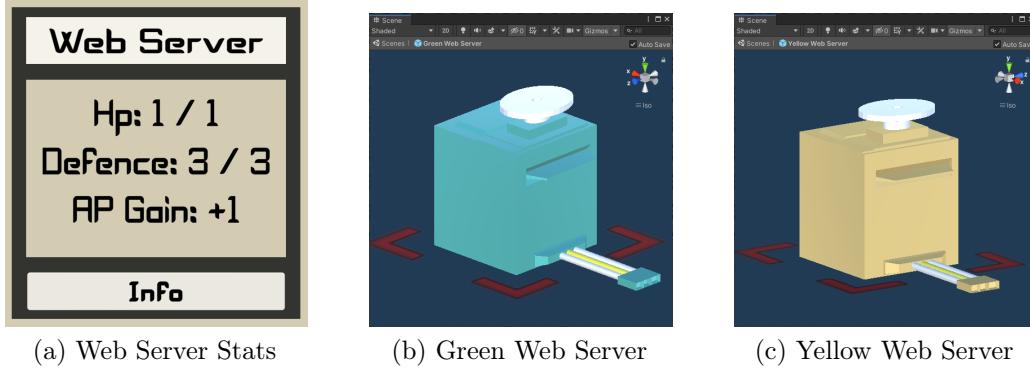


Figure 44: Web Server Stats and Model [55]

The database server is responsible for managing back-end operations of the web application, and allows users to store and retrieve data. Important sensitive information should be kept isolated on a different server such that it cannot be accessed from front-end users.

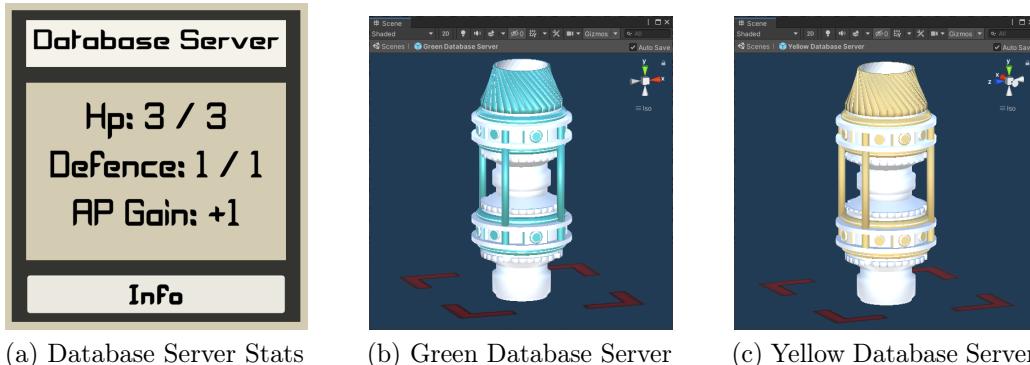


Figure 45: Database Server Stats and Model [56]

Both these server objects generate +1 action points per turn to all other units, unless disabled or destroyed.

5.9 Information Menu

To promote learning within the application, each unit, or ability, has a "info" button which provides more information about an ability or unit's nature. This is recommended as it provides hints about which units an attack vector is extra potent against.

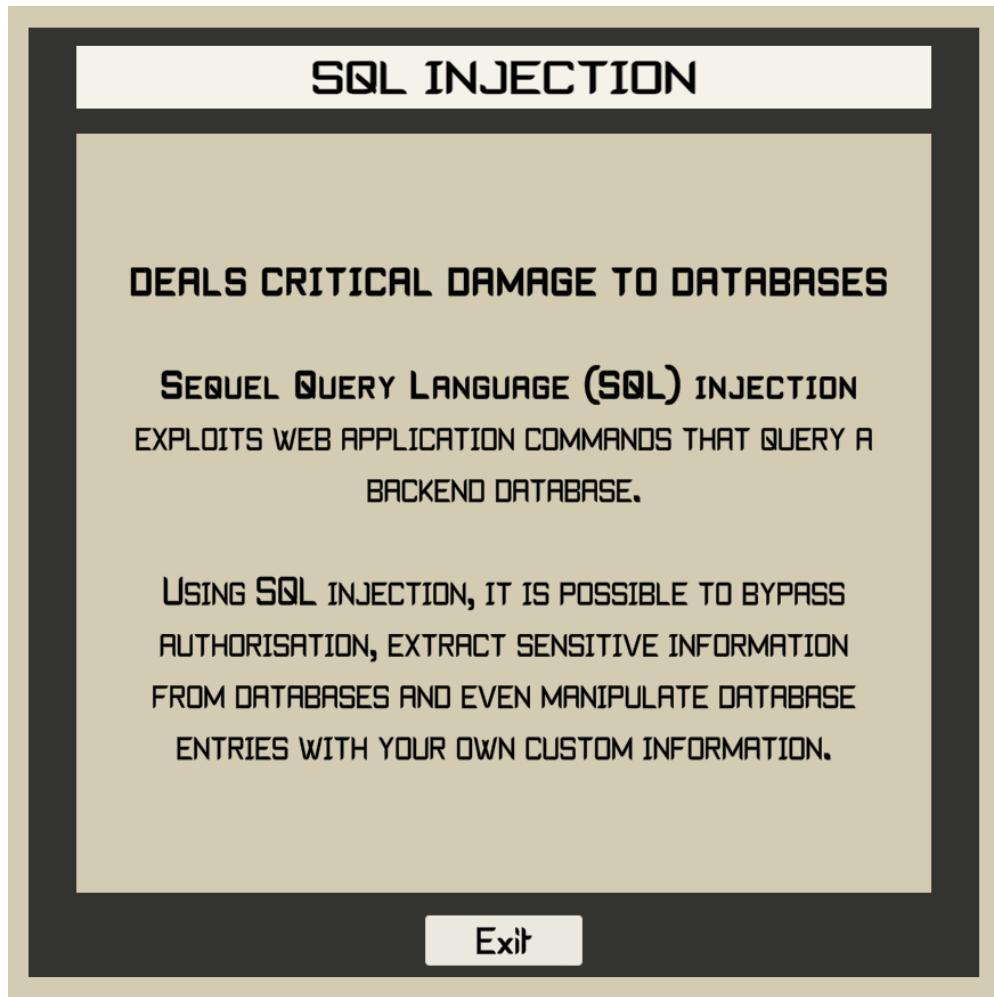


Figure 46: Information Screen Example - SQL Injection

5.10 Status, Hints & Tips Bar

Without implementing an 'in-your-face-tutorial', the application presents a status bar which summarises each mechanic in the game, as well as presenting hints about the nature of each ability.



Figure 47: Status Bar Example - Selecting 'Implement Firewall'

5.11 History Log

To facilitate ease of use and understanding, figure 48 illustrates a typical history log which instructs the user of actions taken, as well as the outcome. Each action taken in the game is appended to the history log of both players with their perspective.

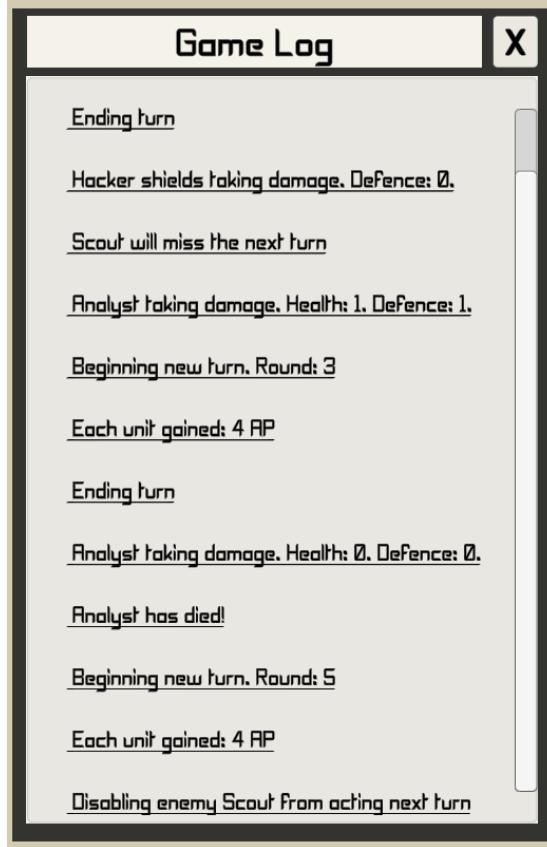


Figure 48: History Log

5.12 Quit Menu

Figure 49 provides the player with the option to resign and safely return to the main connection screen at any point.



Figure 49: Quit Menu

6 Testing and Evaluation

The following section explores the testing strategies implemented which include live debugging (with ParallelSync), unit testing, identifying edges cases and evaluating the performance of the application with the Unity profiler. Alongside Unity profiler, JetBrains Rider was utilised as an integrated development environment for its inbuilt functionality of providing performance recommendations whilst programming.

6.1 Unit Testing

For small single-player games, play-testing in Unity3is typically very easy via simply running the game. However, as a project expands, it can be a very time-consuming process to launch the game, load menus, connect to another player, and perform actions to reach a desired game state to check one thing. As such, automated testing on pure functions is a necessity.

Figure 50 showcases automated tests that were performed on all pure functions within the application (i.e., functions that should always return the same output on a given input). These include using unique moves with a range of parameters, changing the state of selectable units, and the functional correctness of all calculations (such as calculating damage and action points).

6.2 Identifying Edge Cases

Whilst testing these functions, all edge cases were identified and tested (such as negative, positive and zero values). A large benefit of test-driven development is that it enforces good programming practises in that functions only perform one operation, can be debugged easily and flexibly re-written (without breaking other components). A list of these tests can be seen in Appendix A.



Figure 50: Unity Test Runner

6.3 ParallelSync for Live Builds

ParallelSync [57] is a Unity open source extension which clones the Unity development environment. This is fundamentally important for play-testing a multi-client game as you cannot have duplications of the same environment open concurrently. Debugging without would require rebuilding the project externally which can take several minutes per build (namely for WebGL development builds).

ParallelSync works by cloning the folder structure of the project without needing to duplicate all files, assets, and game scenes. Instead it establishes pointers to the original file location with read only access. Because of this, it is safe and easy to create, load and destroy as many clone environments as required.

6.4 Performance Testing

Unity's integrated profiler was utilised to analyse the performance of the application. Since this is a time-consuming process, it is recommended to complete this at the end of a significant feature implementation [58]. Profiling was done comparatively in that sections of the game were recorded, saved, and analytically compared to its previous performance. It is wise to profile after playing a game for an extended amount of time such that application has reached a sustained state [59].

Unity profiling is an instrumentation-based profiler [58]. At each stage, a ‘snippet’ of 300 frames of is captured by inserting a small marker in every function call – and calculating the execution time of this function. Although this creates a tiny bit of overhead, the overall profile is unchanged. Figure 51 demonstrates the overall frame rate captured which sustained a mean average of 7.03ms (142 frames per second).

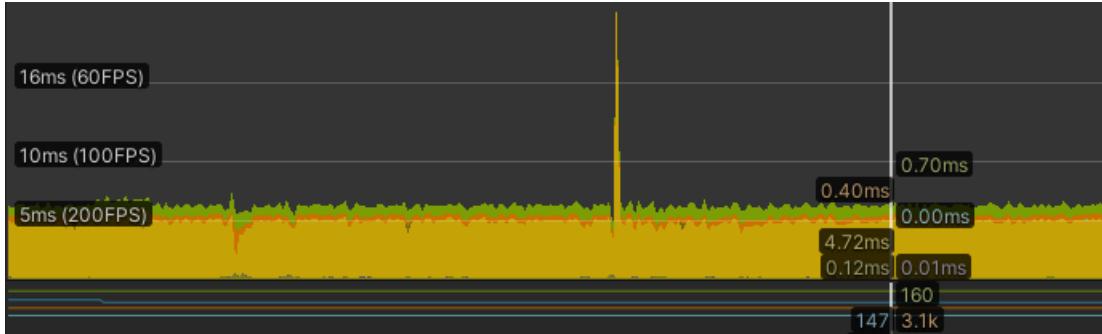


Figure 51: Average FPS of a 300-frame sample

6.5 Achieving a Target Frame Rate

In each frame, the core update loop is processed by the CPU and will require a certain number of milliseconds to complete; this time is inversely proportional to the frame rate [59]. Therefore, to achieve a target frame rate, the core update loop must be less than, or equal to, a target budget which is given by:

$$\text{Budget} = 1000 \text{ ms} / \text{Target Frame Rate}$$

To achieve 60 FPS, the target budget would be equal to: $1000 \text{ ms} / 60 = 16 \text{ ms}$. Therefore, the application's target budget is 16ms - such that each frame within the player's update loop is executed in 16ms or less consistently.

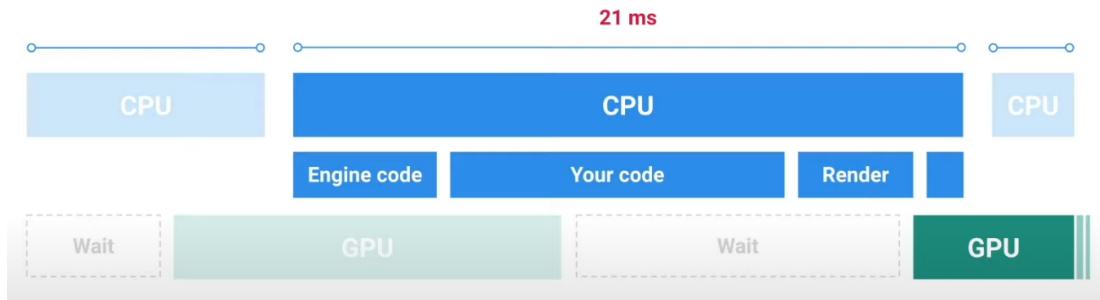


Figure 52: Structure of a Frame [59]

6.6 Evaluation and Optimisation

The first significant optimisation re-worked how tiles updated their status. Since the game scene had 135 tiles, there were 135 tile updates per frame (occupying 15.13% of the players update loop - figure 57, Appendix 11.2). Although this only occupied 0.15ms (0.9% of our target budget), a requirement of the application was to be scalable with map size in new scenarios. For example, if the game was scaled to 1350 tiles, this would become 9% of our target budget! To amend this, tiles were re-worked such that other classes only invoke a tile status change when necessary. This result can be seen in figure 58, Appendix 11.2. One down-side of this approach however is that it couples the tile component to other classes, which was previously completely decoupled.

The second significant optimisation fixed unnecessary memory losses as well as visual bug that caused units to 'shake' within the WebGL version. From profiling it was observed that the garbage collector was invoked 544 times to re-allocate 20kb of memory per frame (figures 53 & 54). This resulted from BFS calculations initiating 493 ray-casts to adjacent tiles each frame. This is significant because the number of ray-casts is proportional to the number of tiles and ray-casting is a relatively expensive operation; at 135 tiles, this was occupying 0.89ms (5.3%) of our target budget. To amend this, tile data was cached upon initialisation and BFS calculations were modified such that they were only invoked once upon selecting a unit.

Assembly-CSharp.dll!SoloDebugging::SoloController.Update() [Invoke] - Total time: 0.89 ms				
Called From	Calls	GC Alloc	Time ms	Time %
Physics.OverlapBox	540	0	0.00	0.00
Physics.Raycast	493	0	0.00	0.00
GC.Alloc	544	21616	0.00	0.00
Called From	Calls	GC Alloc	Time ms	Time %
BehaviourUpdate	1	21616	0.89	76.18

Figure 53: 544 GC Allocations per Frame (21kb)

Used Total: 98.8 MB	Unity: 46.8 MB	Mono: 3.2 MB	GfxDriver: 13.9 MB	Audio: 1.2 MB	Video: 0 B	Profiler: 33.7 MB						
Reserved Total: 153.8 MB	Unity: 98.4 MB	Mono: 4.3 MB	GfxDriver: 13.9 MB	Audio: 1.2 MB	Video: 0 B	Profiler: 36.0 MB						
Total System Memory Usage: 372.0 MB												
Textures: 51 / 77.9 MB												
Meshes: 33 / 117.0 KB												
Materials: 170 / 288.0 KB												
AnimationClips: 0 / 0 B												
AudioClips: 0 / 0 B												
Assets: 628												
GameObjects in Scene: 430												
Total Objects in Scene: 2537												
Total Object Count: 3165												
GC Allocations per Frame: 544 / 21.0 KB												

Figure 54: Memory Usage (per 300 frame capture)

To conclude, after optimising the game, the mean frame budget was lowered to 6.97 (1.69% less per frame), with no notable lag spikes exceeding the target budget of 16ms (see figure 60, Appendix 11.2).

6.7 Evaluation of Networking Strategy

Photon Cloud and PUN2 proved to be incredibly optimised for creating a cross-platform multiplayer application. Regarding performance and frame rate, there was negligible impact in single player mode vs having two multiplayer clients. Figure 55 illustrates the performance impact of using the PUN2 API per 300 frame sample, which had a mean budget of 0.02ms per frame (0.13% of our target). One caveat of Photon Cloud is that matchmaking can only occur within server clusters located in the same region.

Marker Name	Depth	Media	Media	Mean	Min	Max	Range	Count	Count Fr	At Median Frame
PhotonUnityNetworking. 4	0.00		0.01	0.00	0.23	0.23	299	1		0.00
PhotonUnityNetworking. 4	0.00		0.00	0.00	0.04	0.04	1794	6		0.00
PhotonUnityNetworking. 4	0.00		0.01	0.00	0.16	0.16	105	1		0.00

Figure 55: Photon Networking Impact Per 300 Frame Sample

6.8 Evaluation of Application Frame Rate

Table 6 illustrates the average, minimum and maximum frame rate achieved on different platforms. Each test was recorded after 2 minutes to achieve a sustained state.

Platform	Avg FPS	Min FPS	Max FPS
Windows (PC)	144	97	160
Microsoft Edge	136	96	135
Google Chrome	133	109	141
Firefox	71	56	73
Safari (iPad 2017)	42	22	45
Google Chrome*	12	10	24

Table 6: Performance on Different Platforms

From Table 6, it is evident that Google Chrome and Microsoft Edge heavily support WebGL as the performance on these browsers matched that of the Windows application (used as a benchmark). Note these frame rates are only achievable from enabling hardware acceleration which WebGL requires to access the system's dedicated graphics processing unit [41].

The final row in Table 6 demonstrates the frame rate in Google Chrome without hardware acceleration which exemplifies its importance. Unfortunately, Internet Explorer did not support WebGL.

It must be noted that these frame rates are dependent on the hardware available. However, even on weaker devices, 30fps was still achieved which is the industry standard.

6.9 Evaluation of Project Completion

Table 7 presents the completion of all user stories established from the project requirements.

The core goal of the project was successful as all ‘musts’ were completed with exception of task 28. Task 28 refers to how effective the application is at teaching the OWASP security risks which can only be answered from validation testing. User feedback was not prioritised in this project as it was out of scope of the MDA framework. Regarding this however, the Design-Dynamics-Experience framework [60] extends upon this framework by evaluating the user experience from the existing design and dynamics. Unfortunately, this approach requires more time as it adds more components into the overall development process.

All ‘shoulds’ were fully implemented, bar task 18 which was partially implemented through a tips & hints system. A full tutorial implementation was prioritised last in the development cycle as it was identified in the literature review to not be a core necessity.

Regarding the incomplete ‘coulds’, tasks 29 and 30 are befitting for future work. Task 29 stems from the application-requirement of being extensible to creating new scenarios, units, abilities and adding more mechanics in general.

In task 30, it was realised that developing an automated feedback system would have a similar complexity to that used in Chess. However, Chess has a rigorous history and ruleset which this project does not. Therefore, such a feature would not be feasibly possible given the timespan of this project.

ID	functionality or feature	MoSCoW	Completion
1	View a unit’s stats (HP/Defence)	Must	Yes
2	View an ability’s range, damage & cost	Must	Yes
3	View a colour-blind friendly UI	Should	Yes
4	View a unit’s movement range	Must	Yes
5	View a unit’s ability attack range	Must	Yes
6	Move a unit	Must	Yes
7	Attack a target	Must	Yes
8	Disable the target	Must	Yes
9	Defend an ally unit	Must	Yes
10	Click on a move description	Must	Yes
11	Click on a unit	Must	Yes
12	Click on and view a history log	Should	Yes
13	View the units remaining	Should	Yes
14	Pan the camera and/or zoom in	Could	No
15	Unique animations for each ability	Could	No
16	Varied unit abilities	Could	Yes
17	Cost weighting for each ability	Should	Yes
18	An intermediate tutorial	Should	Partial
19	A help / status bar	Should	Yes
20	Multiplayer functionality	Must	Yes
21	Single player functionality	Won’t	N/A
22	Quit / leave the session	Must	Yes
23	Win or lose	Must	Yes
24	Play on Desktop	Must	Yes
25	Play on Web Browser	Should	Yes
26	Play on Tablet	Could	Yes
27	Play cross-platform sessions	Could	Yes
28	Learn about the OWASP #10	Must	Maybe
29	Purchase & upgrade new units	Could	No
30	Receive feedback after a game	Could	No
31	Hear gameplay music	Could	Yes

Table 7: Completion Status of User Stories

7 Project Management

Table 8 describes the tools, techniques and software used to complete this project.

Tools	Description
GitLab	Project management (Boards, Issues and Milestones)
GitHub Desktop	Local software for handling version control
Trello	Lightweight tool for managing daily and weekly tasks
Workona	Extension for organising research into a succinct workplace
Menderley	Reference management for literature research
Microsoft Teams	Communication software for weekly remote meetings
Google Drive	Cloud storage for research documents and recording minutes
Lucid Chart	Online software for UML diagrams
Unity	Game engine for developing for PC and the Web (WebGL)
C#	Primary programming language for Unity
JetBrains Rider	Integrated development environment for C# & Unity
Photon Cloud	Cloud service for hosting multiplayer servers online
PUN2	Multiplayer networking API
ParralelSync	Open source software to clone the Unity development environment
Itch.IO	Secure domain to host WebGL games online

Table 8: Project Management Tools & Techniques

7.1 Reflection of Project Management

The project was completed with the Agile methodology in mind. As such, GitLab's Issues and Milestones were used to deliver weekly updates. This provided invaluable project feedback and assistance which ensured the development process did not deviate.

Despite some personal difficulties in February - March, all major project requirements were completed in accordance to their MoSCoW priority.

7.2 Gantt Chart - Phase One

Table 9 demonstrates the planning and research completed during phase one.

Week Beginning	1	2	3	4	5	6	7	8	9	10
Date beginning	Oct 5	Oct 12	Oct 19	Oct 26	Nov 2	Nov 9	Nov 16	Nov 23	Nov 30	Dec 7
Planning										
Project Brief										
Literature Review										
Submit Project Brief	x									
Project Planning										
Identify Hypothesis & Research Question										
Research										
Review Gamification Mechanics										
Research Cyber Security Games										
Review Related Work										
Identify Project Requirements										
Brain Storm Game										
Write Report										
Submit Report									x	

Table 9: Gantt Chart for Phase One

7.3 Gantt Chart - Phase Two

Tables 10 & 11 present the planned schedule and the actual outcome during phase 2. The ‘implementation’ tasks were adapted from the user stories (requirements) and scheduled in accordance to their MoSCoW priority, as well as their implementation priority. For example, units are required before unit abilities can be implemented.

Week Beginning	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Date beginning	14/12	21/12	28/12	4/1	11/1	18/1	25/1	11/2	28/2	21/2	22/2	21/3	38/3	315/3	322/3	329/3	35/4	12/4	19/4	26/4
Implementation																				
Mock UI Wireframes																				
Set Up Dev Envrioment																				
Multiplayer Networking																				
Design Board Outline																				
Add Movement Logic																				
Add Unit Assets																				
Add Unit Behaviour																				
Add UI Elements																				
Add Unit Abilities																				
Map UI Elements																				
Add Sound Assets																				
Add Tutorial																				
Testing & Evaluation																				
Set Up Unit Tests																				
Desktop Live Build																				
WebGL Live Build																				
Optimise Game																				
Obtain User Feedback																				
Write Final Report																				
Submit Final Report																		x		

Table 10: Planned Gantt Chart for Phase Two

Week Beginning	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Date beginning	14/12	21/12	28/12	4/1	11/1	18/1	25/1	11/2	28/2	21/2	22/2	21/3	38/3	315/3	322/3	329/3	35/4	12/4	19/4	26/4	3/5	10/5
Implementation																					Extension	
Mock Up UI Wireframes																						
Set Up Dev Envrioment																						
Multiplayer Networking																						
Design Board Outline																						
Add Movement Logic																						
Add Unit Assets																						
Add Unit Behaviour																						
Add UI Elements																						
Add Unit Abilities																						
Map UI Elements																						
Add Sound Assets																						
Add Tutorial																						
Testing & Evaluation																						
Desktop Live Build																						
Set Up Unit Tests																						
WebGL Live Build																						
Optimise Game																						
Obtain User Feedback																						
Write Final Report																						
Submit Final Report																					x	

Table 11: Retrospective Gantt Chart for Phase Two

7.4 Risk Assessment

The following risk assessment ensured appropriate actions were taken for the successful completion of this project. The risk exposure (R.E) is given by the product of the probability and severity of each risk.

Risk	Prob (1-5)	Sev (1-10)	R.E	Mitigation
Project deadlines not met	3	10	30	Establish soft deadlines and meet with supervisor weekly for continuous feedback & self-evaluation
Online multiplayer failure (limitations/pricing of cloud server hosting)	3	9	27	Develop LAN multiplayer functionality from Unity Mirror API, or local multiplayer co-op
Application not effective at teaching	3	7	21	By identifying an appropriate target demographic and the most appropriate cyber security content to teach
Over-estimating scope of implementation	3	7	21	Plan project into smaller iterations and prioritise according to MoSCoW
Application does not relate to original problem statement & hypothesis	2	9	18	Continuously referring to the initial problem statement and hypothesis
Application does not meet user requirements	2	8	16	Prioritise ‘musts’, ‘shoulds’ and then ‘coulds’
Sickness, flu or mental health difficulties	2	8	16	Exercising daily, eating healthy, breaking down tasks into small achievable chunks, and reach out for support if needed
Impact of Covid19	5	3	15	Application will utilise online multiplayer (for remote gameplay), alongside Microsoft Teams, Discord & GitLab for communication and development
Gamified mechanics not implemented appropriately	2	7	14	By identifying key mechanics through literature and game review, and use the MDA framework to decide upon the decided mechanics for the game
Stolen work/data from: cloud storage being compromised & ransomware	1	10	10	Using a secure password manager, 2FA and manually backing up important files on multiple locations
Loss of work/data	1	10	10	Git version control and OneDrive/Google Drive for cloud storage
Under-estimating scope of implementation	1	5	5	Further refine/iterate the game after evaluation, and implement all ‘coulds’ if time permits

Table 12: Risk Assessment

8 Future Work

Additional Scenarios

Additional scenarios and levels would provide more out of the box experiences and interactivity with the application.

Custom Level Editor

A custom level editor (UI) would allow players to modify units, ability types and the level design, without Unity, to create their own game-scenarios for the purpose of training.

Upgradable Units & Abilities

An upgrade skill tree for unit abilities or unit stats would add an extra depth of strategy to the pre-existing mechanics.

Evaluation from Feedback

To test if learning objectives are met, receiving feedback on the existing design and dynamics would be important to verify the applications effectivity, and improve it in doing so.

Single-player Mode & Story

A single-player story with an overall objective would engage users who cannot play multiplayer.

Additional Gamification Mechanics

Incorporating feedback through an automated system based on actions taken could provide more educational value. Alternatively, in game quizzes could be set between turns which test a player's knowledge to receive a strategic advantage (or disadvantage) upon answering.

9 Conclusion

This project set out with the following research question and hypothesis:

Research Question

Can teaching cyber security through a gamified medium improve cyber security awareness?

Hypothesis

Using the MDA framework will satisfy the educational requirements for creating an serious game

Regarding the research question, through a thorough review of online accessible games, related work and supporting literature review, yes gamified strategies can improve cyber security awareness.

Regarding the hypothesis, implementing the MDA framework with a variety of mechanics identified in Forde's framework, and from literature review, creating an educational game was possible. However, this was not without its challenges as this project had many branch points in which an idea for one mechanic or theme wanted to bloom - and trying to find a focused balance to complete a project in a set time was rather challenging. Because designing good games is hard in general, with fun educational games being even more challenging, a project like this ultimately needs a more flexible schedule such that each iteration of design can be periodically reviewed and tested from user feedback, which again is exemplified by the Design, Dynamics, Experience model [60].

Regarding the overall goals and completion status, this project was successful in that all 'musts' and 'shoulds completed, as well as the most important 'coulds'. A minimum viable product was still achieved which can be played above 30FPS on all platforms tested, with 130FPS being the average for Google Chrome, Microsoft Edge and PC. Furthermore, the multiplayer aspect of this game can be accessed at any time without hardware utilisation, as well support for cross-platform matchmaking. Since the game was developed with mobile controls in mind, the game can be fully played even with touch screen controls. Furthermore, since the game is based off a well - known game archetype (Chess), it should be intrinsically familiar with any target audience irrespective of video gaming backgrounds (as demonstrated in Program Wars [11]).

Although only one scenario was created and exemplified (using the OWASP Top Ten [30]), by using a neat component based design methodology with 'prefabs' for all core mechanics, units, abilities and maps, any user could use the project as a framework to build their own game without any programming requirement. This is because all parameters for unit stats, OWASP information, names can be freely changed within the inspector in such a way that the scripts interpret this information and parse it correctly to other UI elements and calculations. This is also true for the unit abilities. To summarise, only the aesthetics & and physical

designs need to be altered.

Now that the framework for a 2-player, cyber security theme game has been completed, the next logical step would be to re-evaluate the existing design and dynamics utilising the DDE framework. This would ensure that educational aspects are appropriately tested (which was a main criticism of Control-Alt-Hack [12], and not sure part of an overall educationally themed game.

10 Bibliography

- [1] M. Bada and J. R. Nurse, “Developing cybersecurity education and awareness programmes for small-and medium-sized enterprises (smes),” *Information & Computer Security*, 2019.
- [2] D. Buil-Gil, F. Miró-Llinares, A. Moneva, S. Kemp, and N. Díaz-Castaño, “Cybercrime and shifts in opportunities during covid-19: a preliminary analysis in the uk,” *European Societies*, pp. 1–13, 2020.
- [3] H. S. Lallie, L. A. Shepherd, J. R. Nurse, A. Erola, G. Epiphanou, C. Maple, and X. Bellekens, “Cyber security in the age of covid-19: a timeline and analysis of cyber-crime and cyber-attacks during the pandemic,” *arXiv preprint arXiv:2006.11929*, 2020.
- [4] J.-N. Tioh, M. Mina, and D. W. Jacobson, “Cyber security training a survey of serious games in cyber security,” in *2017 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2017, pp. 1–5.
- [5] A. T. Forde, “A gamification toolkit for improving cyber security standards adoption,” Master’s thesis, University of Southampton, September 2020.
- [6] R. Hunicke, M. LeBlanc, and R. Zubek, “Mda: A formal approach to game design and game research,” in *Proceedings of the AAAI Workshop on Challenges in Game AI*, vol. 4, no. 1. San Jose, CA, 2004, p. 1722.
- [7] H. Aldawood and G. Skinner, “Reviewing cyber security social engineering training and awareness programs—pitfalls and ongoing issues,” *Future Internet*, vol. 11, no. 3, p. 73, 2019.
- [8] J. Abawajy, “User preference of cyber security awareness delivery methods,” *Behaviour & Information Technology*, vol. 33, no. 3, pp. 237–248, 2014.
- [9] S. Hart, A. Margheri, F. Paci, and V. Sassone, “Riskio: A serious game for cyber security awareness and education,” *Computers & Security*, p. 101827, 2020.
- [10] A. Jøsang, V. Stray, and H. Rygge, “Threat poker: Gamification of secure agile,” in *IFIP World Conference on Information Security Education*. Springer, 2020, pp. 142–155.
- [11] J. Anvik, V. Cote, and J. Riehl, “Program wars: a card game for learning programming and cybersecurity concepts,” in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 2019, pp. 393–399.
- [12] T. Denning, A. Lerner, A. Shostack, and T. Kohno, “Control-alt-hack: the design and evaluation of a card game for computer security awareness and education,” in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 915–928.

- [13] “Game of Threats,” Accessed: 14-12-2020. [Online]. Available: <https://www.pwc.co.uk/issues/cyber-security-services/game-of-threats.html>
- [14] “Cybersecurity Lab - NOVA LABS,” Accessed: 14-12-2020. [Online]. Available: <https://www.pbs.org/wgbh/nova/labs/lab/cyber/>
- [15] “Webonauts Internet Academy,” Accessed: 14-12-2020. [Online]. Available: <https://learningworksforkids.com/apps/webonauts-internet-academy/>
- [16] “Targeted Attack - The Simulation,” Accessed: 14-12-2020. [Online]. Available: <http://targetedattacks.trendmicro.com/>
- [17] “Keep Tradition Secure,” Accessed: 14-12-2020. [Online]. Available: <https://keeptraditionsecure.tamu.edu/>
- [18] “Cyber Awareness Challenge USA,” Accessed: 14-12-2020. [Online]. Available: <http://sgschallenge.com/cyber-awareness-challenge/>
- [19] “Cyberland - Cyber Security Challenge UK,” Accessed: 14-12-2020. [Online]. Available: <https://www.cybersecuritychallenge.org.uk/what-we-do/schools-programme/cyberland>
- [20] “Hacknet,” Accessed: 14-12-2020. [Online]. Available: <https://hacknet-os.com/>
- [21] “Classcraft,” Accessed: 14-12-2020. [Online]. Available: <https://www.classcraft.com/>
- [22] R. Roepke and U. Schroeder, “The problem with teaching defence against the dark arts: A review of game-based learning applications and serious games for cyber security education.” in *CSEDU* (2), 2019, pp. 58–66.
- [23] M. Thompson and H. Takabi, “Effective of using card games to teach threat modeling for secure web application developments,” *Issues in Information Systems*, vol. 17, no. 3, 2016.
- [24] “OWASP Project Cornucopia,” 2013, Accessed: 03-05-2021. [Online]. Available: <https://owasp.org/www-project-cornucopia/>
- [25] M. Thompson and H. Takabi, “Effectiveness of using card games to teach threat modeling for secure web application developments.” *Issues in Information Systems*, vol. 17, no. 3, 2016.
- [26] C. Koepke, *Persona Image One*. Surface, 2021, Accessed: 19-04-2021. [Online]. Available: <https://unsplash.com/photos/E9NcsvbRVqo>
- [27] Chirstina, *Persona Image Two*. Wocintechat.com, 2021, Accessed: 19-04-2021. [Online]. Available: <https://unsplash.com/photos/S3GrMiUhNU>

- [28] S. Northcutt, *Persona Image Three*. Unsplash, 2021, Accessed: 19-04-2021. [Online]. Available: <https://unsplash.com/photos/sgZX15Da8YE>
- [29] L. Parter, “WebGL 2.0 Specification,” WebGL, 2020, Accessed: 01-05-2021. [Online]. Available: <https://www.khronos.org/registry/webgl/specs/latest/2.0/>
- [30] “Owasp top 10 web application security risks,” 2017, Accessed: 20-04-2021. [Online]. Available: <https://owasp.org/www-project-top-ten/>
- [31] G. Miranda, “Unity tutorial - tactics movement,” 2017, Accessed: 20-04-2021. [Online]. Available: <http://www.gameprogrammingacademy.com/unity-tutorial-tactics-movement/>
- [32] “Colour Wheel Calculator,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://www.sessions.edu/color-calculator/>
- [33] “Web Content Accessibility Guidelines,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://www.w3.org/TR/2018/REC-WCAG21-20180605/>
- [34] “Font Contrast Checker,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://webaim.org/resources/contrastchecker/?fcolor=000000&bcolor=D3CCB3>
- [35] “Font Contrast Checker,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://webaim.org/resources/contrastchecker/?fcolor=000000&bcolor=F5F2E9>
- [36] “Colour blindness awareness,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://www.colourblindawareness.org/>
- [37] D. Nichols, “Coloring for colorblindness,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://davidmathlogic.com/colorblind/#%23E1BE6A-%2340B0A6>
- [38] “Doodad Pattern Generator,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://doodad.dev/pattern-generator/>
- [39] D. Baron, *Hands-On Game Development Patterns with Unity 2019: Create engaging games by using industry-standard design patterns with C.* Packt Publishing Ltd, 2019.
- [40] “Unity 3d,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://unity.com/>
- [41] “Unity - manual: Getting started with webgl development,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://docs.unity3d.com/Manual/webgl-gettingstarted.html>
- [42] “Phaser3,” 2021, Accessed: 19-04-2021. [Online]. Available: <https://phaser.io/phaser3>

- [43] “Photon cloud vs photon server,” 2021, Accessed: 19-04-2021. [Online]. Available:
<https://doc.photonengine.com/en-us/realtimetime/current/getting-started/onpremises-or-saas>
- [44] “Pun2 documentation,” 2021, Accessed: 19-04-2021. [Online]. Available:
<https://doc-api.photonengine.com/en/PUN/v2/index.html>
- [45] “Mirror networking,” 2021, Accessed: 19-04-2021. [Online]. Available:
<https://mirror-networking.gitbook.io/docs/>
- [46] “Rpcs and raiseevents,” 2021, Accessed: 19-04-2021. [Online]. Available:
<https://doc.photonengine.com/en-us/pun/v2/gameplay/rpcsandraiseevent>
- [47] Foodylag, “Spider Robot,” OpenGameArt.org, 2017, Accessed: 28-04-2021. [Online]. Available: <https://opengameart.org/content/spider-robot>
- [48] “Owasp top 10 web application security risks,” 2017, Accessed: 03-05-2021. [Online]. Available:
<https://www.synopsys.com/glossary/what-is-owasp-top-10.html>
- [49] G. Svaiko, “The State of Security,” *Tripwire*, pp. 1–13, 2020. [Online]. Available:
<https://www.tripwire.com/state-of-security/featured/most-common-website-security-attacks-and>
- [50] Lacomap, “Droid,” Blendswap.com, 2017, Accessed: 28-04-2021. [Online]. Available: <https://www.blendswap.com/blend/19351>
- [51] Z. Shaheen, “Portal 2 Weatley,” Blendswap.com, 2015, Accessed: 28-04-2021. [Online]. Available: <https://www.blendswap.com/blend/15053>
- [52] Galbatorix123, “Cam Bot,” Blendswap.com, 2014, Accessed: 28-04-2021. [Online]. Available: <https://www.blendswap.com/blend/13529>
- [53] “Web application firewall,” 2017, Accessed: 03-05-2021. [Online]. Available:
https://owasp.org/www-community/Web_Application_Firewall
- [54] “2017 broken access control,” 2017, Accessed: 03-05-2021. [Online]. Available:
https://owasp.org/www-project-top-ten/2017/A5_2017-Broken_Access_Control
- [55] Rollerrabbit2, “Servomotor Simple,” Blendswap.com, 2017, Accessed: 28-04-2021. [Online]. Available: <https://www.blendswap.com/blend/19199>
- [56] Reaper2k, “Sci-Fi Thing,” Blendswap.com, 2015, Accessed: 28-04-2021. [Online]. Available: <https://blendswap.com/blend/14463>
- [57] “Parralelsync - github,” 2021, Accessed: 19-04-2021. [Online]. Available:
<https://github.com/VeriorPies/ParrelSync>
- [58] “Unity - manual: The profiler window,” 2021, Accessed: 19-04-2021. [Online]. Available:
<https://docs.unity3d.com/Manual/ProfilerWindow.html>

- [59] C. Continisio, “Introduction to profiling in Unity 2020,” YouTube, 2020, Accessed: 28-04-2021. [Online]. Available:
https://www.youtube.com/watch?v=XRURWwabF4&t=431s&ab_channel=Unity
- [60] W. Walk, D. Görlich, and M. Barrett, “Design, dynamics, experience (dde): an advancement of the mda framework for game design,” in *Game Dynamics*. Springer, 2017, pp. 27–45.

11 Appendices

11.1 Unit Test Cases

Current Defence	Max Defence	Defence Gain	Expected Defence
0	3	0	0
0	3	1	1
1	3	2	3
2	3	3	3
0	0	1	0
3	3	-1	3

Table 13: Testing 'Restore Shields' Function

Defence	Health	Damage	Expected Health	Expected Defence
0	0	0	0	0
0	1	0	0	0
0	1	1	0	0
0	2	1	1	0
1	0	1	0	0
1	1	1	1	0
1	2	2	1	0
2	0	3	0	0
1	1	-1	1	1

Table 14: Testing 'Take Damage' Function

Defence	Health	Damage	Expected Health	Expected Defence
0	0	0	0	0
1	2	1	1	1
2	2	2	0	2
3	2	3	0	3
3	2	-1	2	3

Table 15: Testing 'Bypass Shields' Function

Action Points	Max AP	AP Requirement	Expected Return
3	6	-10	FALSE
3	6	-5	FALSE
3	6	0	TRUE
3	6	2	TRUE
3	6	3	TRUE
3	6	10	FALSE

Table 16: Check Ability AP Requirement

Action Points	Max AP	AP Increment	Expected AP
3	6	-10	3
3	6	-5	3
3	6	0	3
3	6	2	5
3	6	3	6
3	6	10	6

Table 17: Testing 'Increment AP' Function

Action Points	Max AP	AP Decrement	Expected AP
3	6	-10	3
3	6	-5	3
3	6	0	3
3	6	2	1
3	6	3	0
3	6	10	0

Table 18: Testing 'Decrement AP' Function

11.2 Performance Profiling in Unity3D

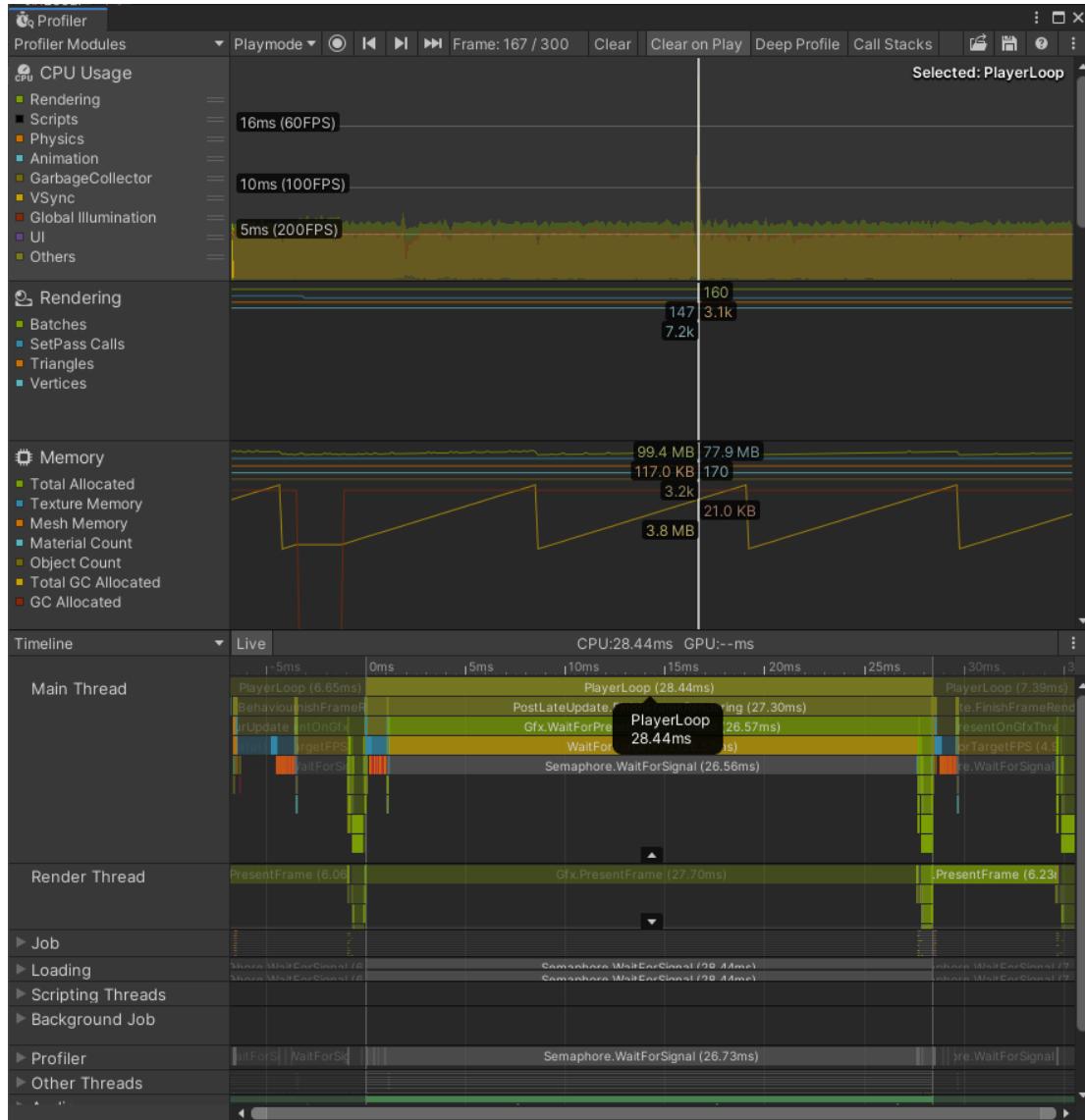


Figure 56: Initial Performance with a Mean Budget (7.09) and Lag Spike of 28.44ms

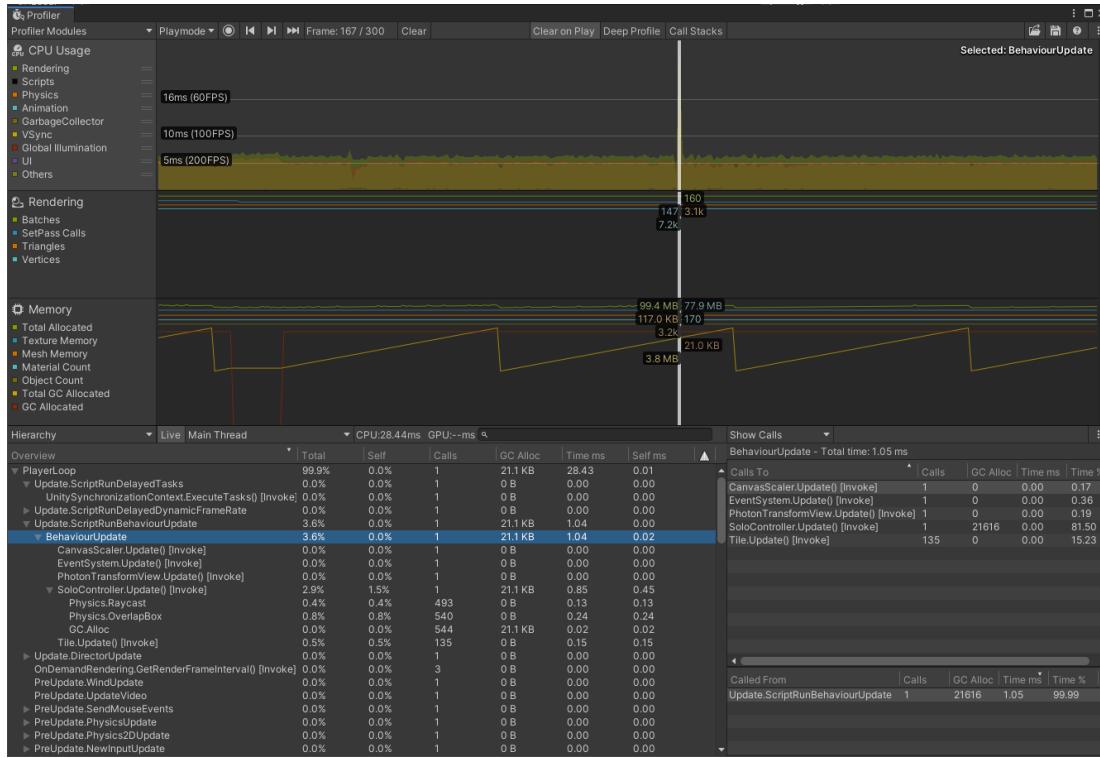


Figure 57: 135 Tile Calls Occupying 15% of the Player's Update Loop

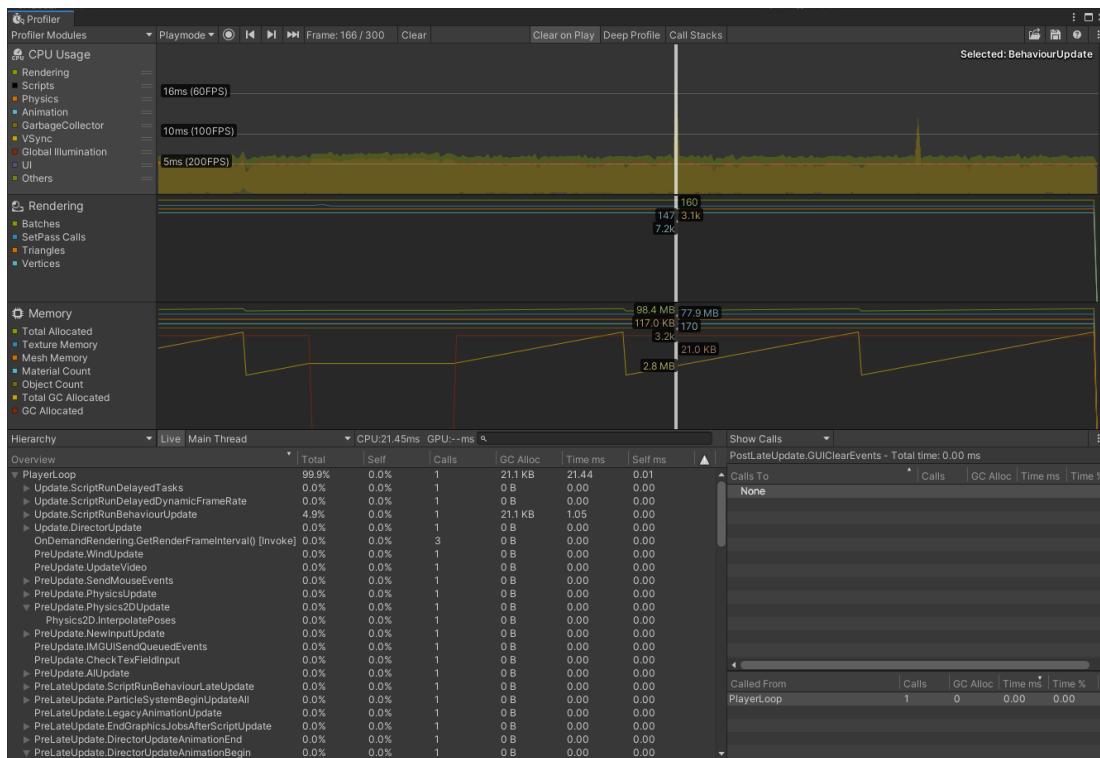


Figure 58: Reduction of lag spike to 21.44ms

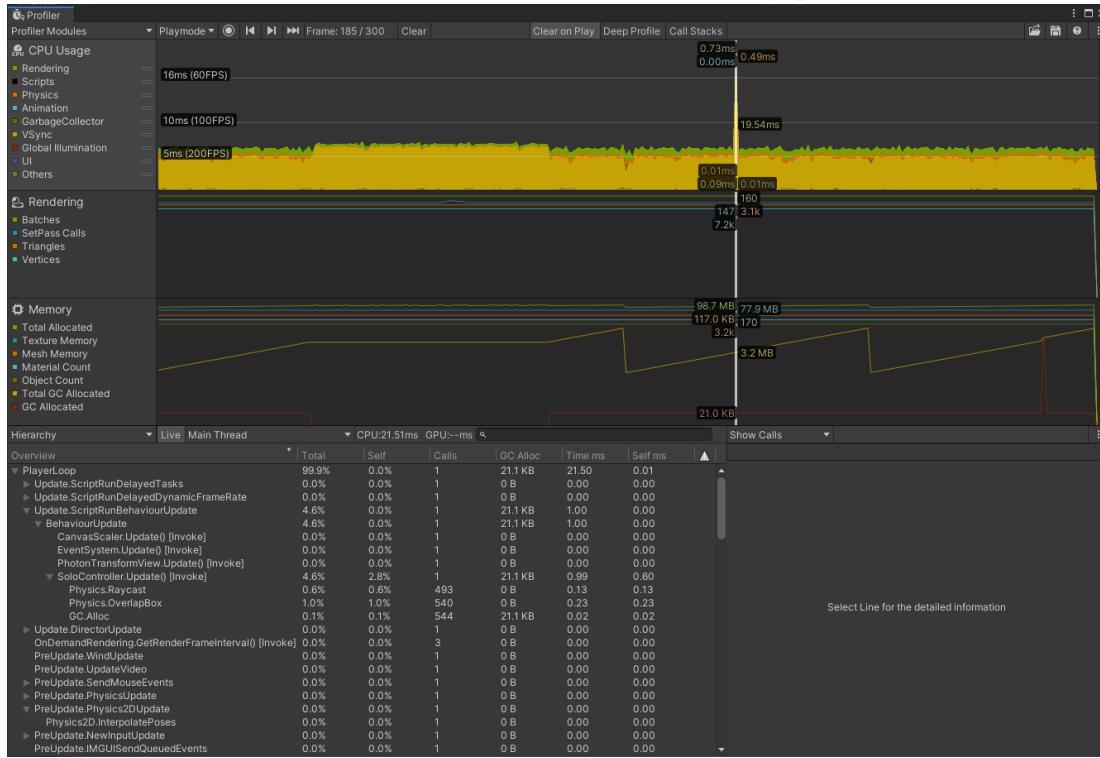


Figure 59: Performance Improvement after Removing Tiles from the Update Loop

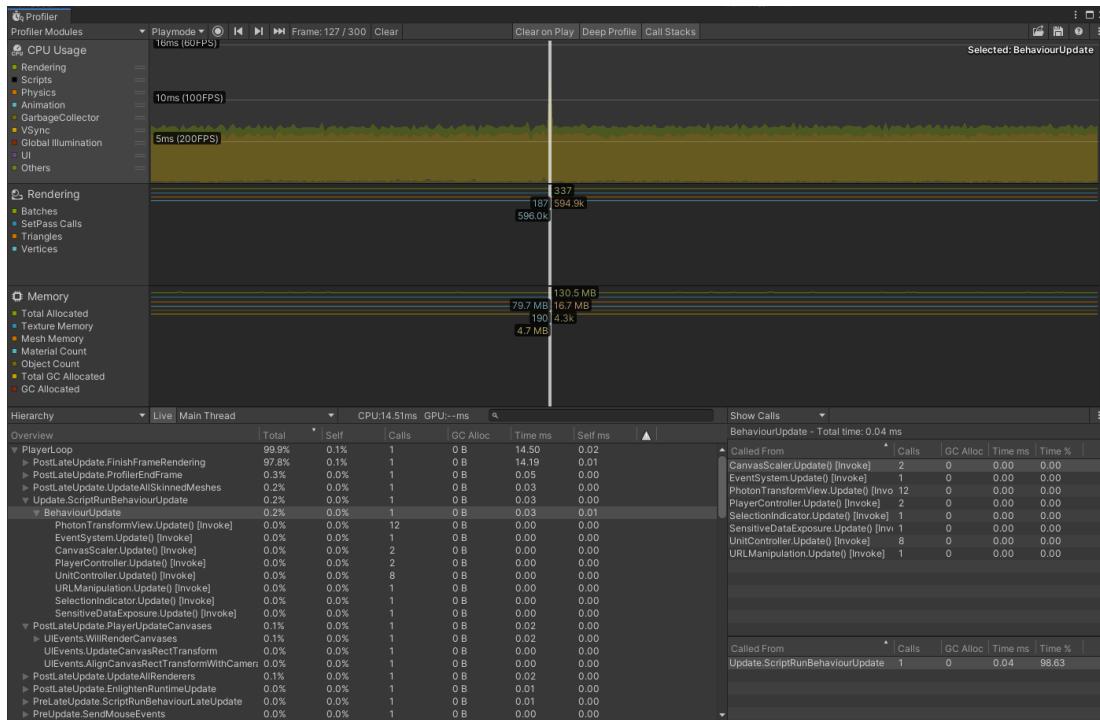


Figure 60: Performance Improvement from Removing Unnecessary Ray-casts

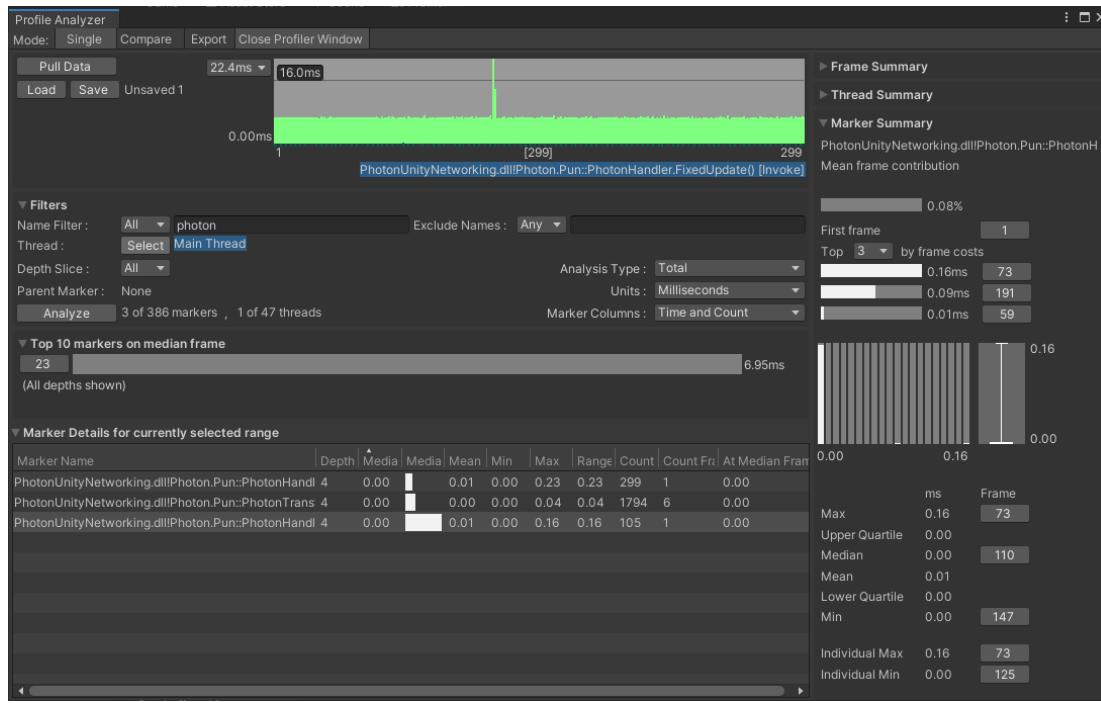


Figure 61: Negligible Performance Impact of Networking in WebGL Build